7-11-91

IN-07

2851

p.116

NASA Contractor Report CR-187105

Investigation of Advanced Counterrotation Blade Configuration Concepts for High Speed Turboprop Systems

Task II - Unsteady Ducted Propfan Analysis
Computer Program Users Manual

Edward J. Hall, Robert A. Delaney, and James L. Bettner
Allison Gas Turbine Division of General Motors
Indianapolis, Indiana

May 1991

NASA

Prepared for
Lewis Research Center
Under Contract NAS3-25270

Preface

This manual was prepared by Edward J. Hall, Robert A. Delaney, and James L. Bettner of the Allison Gas Turbine Division, General Motors Corporation, Indianapolis, IN. The work was performed under NASA Contract NAS3-25270 from March, 1990 to March, 1991. The grid generation, flow code theory, and programming modifications necessary for the analysis of ducted propfans were performed by Edward J. Hall. The Allison program manager for this contract was James L. Bettner. The NASA program manager for this contract was Christopher J. Miller.

Acknowledgements

The authors would like to express their appreciation to the following NASA personnel who contributed to this program:

Mark Celestina for his helpful suggestions concerning the development of the viscous solver,

Dr. John. J. Adamczyk for the many helpful technical discussions concerning the development of the computer codes,

Dr. Christopher J. Miller for his suggestions and critical review of the program,

Dr. David P. Miller for the use of his grid generation code, *TIGGERC*,

and Christopher E. Hughes for his helpful comments and suggestions.

The services of the NASA Numerical Aerodynamic Simulation (NAS) facilities and personnel are gratefully acknowledged.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

NOTATION

A list of the symbols used throughout this document and their definitions is provided below for convenience.

Roman Symbols

c_p . . . specific heat at constant pressure

c_v . . . specific heat at constant volume

e . . . total internal energy

i . . . z index of numerical solution

j . . . r index of numerical solution

k . . . θ index of numerical solution or thermal conductivity

n . . . rotational speed (revolutions per second)

r . . . radius or radial coordinate

u_x . . . velocity in the Cartesian x direction

u_y . . . velocity in the Cartesian y direction

u_z . . . velocity in the Cartesian z direction

x . . . Cartesian coordinate system coordinate

y . . . Cartesian coordinate system coordinate

z . . . cylindrical coordinate system axial coordinate

$ADPAC$ ... Advanced Ducted Propfan Analysis Codes

AOA ... Angle of attack aerodynamic analysis code

$AOAPLOT$ ... Ducted propfan automated plotting program

$ASCII$ ... American Standard Code for Information Interchange

B ... number of propeller blades

CFL ... Courant-Freidrichs-Lewy number ($\Delta t/\Delta t_{max,stable}$)

$CHGRIDV2$ ... Ducted propfan grid generation code

D ... Propfan diameter (units of length)

$FULLPLOT$ ... Ducted propfan PostScript x-y plotting program

J ... advance ratio ($J = U/nD$)

M ... Mach number

R ... gas constant or residual or maximum radius

$ROTCGRID$ ... Ducted propfan full rotor grid rotation program

$ROTCFLOW$ ... Ducted propfan full rotor flow rotation program

$SDBLIB$ ... Scientific DataBase Library (binary file I/O routines)

U ... Freestream or flight velocity (units of length/time)

V ... volume

Greek Symbols

β ... local blade setting angle

$\beta_{3/4}$ ... 3/4 radius propfan blade setting angle

γ ... specific heat ratio

Δt ... calculation time step

ρ ... density

$\mu \ldots$ coefficient of viscosity

Subscripts

$[\ ]_{i,j,k} \cdots$ grid point index of variable

$[\ ]_{max} \ldots$ maximum value

$[\ ]_{min} \cdots$ minimum value

1. SUMMARY

The primary objective of this study was the development of a time-dependent three-dimensional Euler/Navier-Stokes aerodynamic analysis to predict unsteady compressible transonic flows about ducted and unducted propfan propulsion systems at angle of attack. The computer codes resulting from this study are referred to as *ADPAC* (Advanced Ducted Propfan Analysis Codes). This report is intended to serve as a computer program user's manual for the *ADPAC* codes developed under Task II of NASA Contract NAS3-25270, Unsteady Ducted Propfan Analysis.

Aerodynamic calculations were based on a four-stage Runge-Kutta time-marching finite volume solution technique with added numerical dissipation. A time-accurate implicit residual smoothing operator was utilized for unsteady flow predictions. For unducted propfans, a single H-type grid was used to discretize each blade passage of the complete propeller. For ducted propfans, a coupled system of five grid blocks utilizing an embedded C-grid about the cowl leading edge were used to discretize each blade passage. Grid systems were generated by a combined algebraic/elliptic algorithm developed specifically for ducted propfans. Numerical calculations were compared with experimental data for both ducted and unducted propfan flows. The solution scheme demonstrated efficiency and accuracy comparable with other schemes of this class.

2

2. INTRODUCTION

This document contains the Computer Program User's Manual for the *ADPAC* (Advanced Ducted Propfan Analysis Codes) 3-D Euler/Navier-Stokes and grid generation analyses developed by the Allison Gas Turbine Division of the General Motors Corporation under Task II of NASA Contract NAS3-25270. The objective of this study was to develop a three-dimensional time-dependent Euler/Navier-Stokes analysis for high-speed ducted propfan aircraft propulsion systems operating at angle of attack. This analysis consists of a grid generation scheme coupled with an advanced aerodynamic analysis code. The grid generation scheme is referred to as *ADPAC-CHGRIDV2* or simply *CHGRIDV2*. The aerodynamic analysis is referred to as *ADPAC-AOA* or simply *AOA*.

AOA utilizes a finite volume multiple-block four-stage Runge-Kutta numerical algorithm to predict the aerodynamics of ducted fan flows. The ability to accurately predict the unsteady aerodynamics due to angle of attack, and the complicated viscous flow interactions between the rotating fan and the cowl were of particular interest in this program. The multiple grid block arrangement simplifies the calculation of the full rotor geometry for angle of attack flows, and permits some unique grid arrangements for complicated ducted propfan geometries. Unducted propfans were analyzed using a single sheared H-type grid for each blade passage. The analysis for ducted propfans was based on a numerically coupled multiple-block grid arrangement with a body-centered C-type grid about the cowl, surrounded by four H-type grid blocks for each blade passage. An illustration of the construction of the multiple-block grid system and the grid block numbering convention for a ducted propfan are given in Fig. 2.1. A detailed theoretical derivation of the analyses and a demonstration of results from these codes are given in the Final Report for this task [1].

To predict the flow about a ducted propfan at angle of attack using the analyses described in this document, the necessary sequence is:

1. Define the geometry.
2. Generate a numerical grid for the domain of interest using *CHGRIDV2*.
3. Run the Euler/Navier-Stokes code *AOA* to predict the steady aerodynamics.
4. Duplicate the single passage steady flow solution for a full rotor configuration (see description of *ROTCGRID* and *ROTCFLOW* in Chapter 5).
5. Run the Euler/Navier-Stokes code *AOA* to predict the unsteady aerodynamics.
6. Plot and process the results as needed using *AOAPLOT* (see Chapter 5) or other codes.

The intermediate calculation of the steady state solution and subsequent rotation of this data for the full rotor (Steps 3 and 4) are actually unnecessary, but serve to reduce the overall computation time for the unsteady solution by providing a reasonable set of initial data from which the time-dependent calculation may begin. A description of the commands involved in the steps described above beginning with

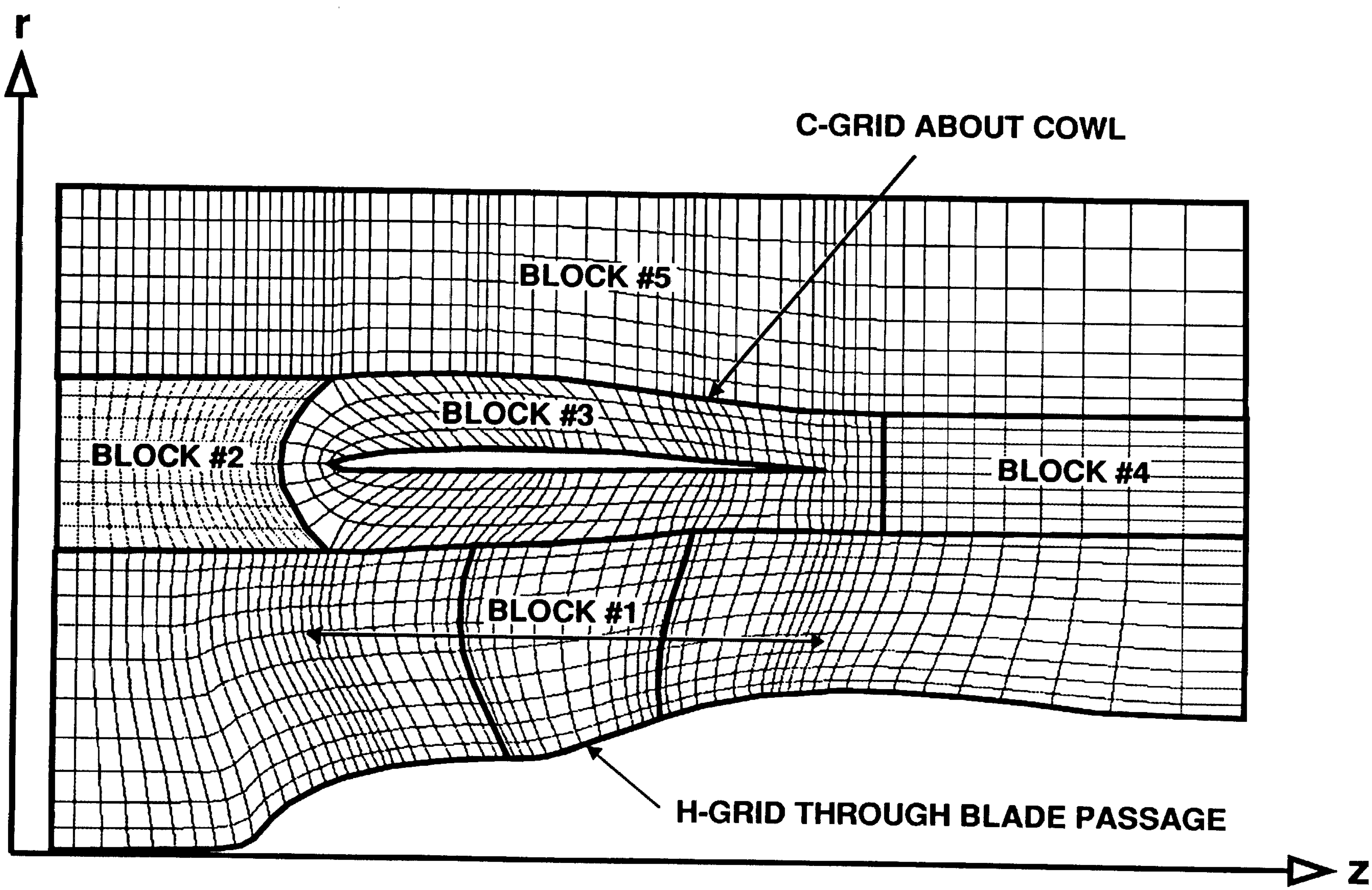

Figure 2.1: Meridional plane view of a ducted propfan multiple-block grid and grid block numbering convention.

the source code distribution, extracting the source files, compiling the codes, running a steady state test case, and, finally, running a time-dependent test case, is given in the Appendix. Separate sections are provided in the chapters which follow to describe the basis and operation of the codes used in the steps above.

It is worthwhile mentioning that the development and application of the codes described in this manual were performed on Unix-based computers. All files are stored in machine-independent format. Small files utilize standard ASCII format, while larger files, which benefit from some type of binary storage format, are based on the Scientific DataBase Library (SDBLIB) format [2]. The SDBLIB format utilizes machine-dependent input/output routines which permit machine independence of the binary data file. The SDBLIB routines are under development at the NASA Lewis Research Center.

Most of the plotting and graphical postprocessing of the solutions was performed on graphics workstations. The *PLOT3D* [3], *SURF* [4], and *FAST* [5] graphics software packages developed at NASA Ames Research Center were extensively used for this purpose, and data files for these plotting packages are generated automatically. These data files are written in what is known as *PLOT3D* multiple-grid format. The format of a *PLOT3D* mesh and flow file are given by the sample FORTRAN coding below:

PLOT3D Mesh File Format FORTRAN Coding Example

```
WRITE( ) MG
WRITE( ) (IL(L), JL(L), KL(L),L=1,MG)
WRITE( ) (((X(I,J,K,L),I=1,IL(L)),J=1,JL(L)),K=1,KL(L)),
```

```
     .          (((Y(I,J,K,L),I=1,IL(L)),J=1,JL(L)),K=1,KL(L)),
     .          (((Z(I,J,K,L),I=1,IL(L)),J=1,JL(L)),K=1,KL(L))
```

PLOT3D Flow File Format FORTRAN Coding Example

```
      WRITE( ) MG
      WRITE( ) (IL(L), JL(L), KL(L),L=1,MG)
      DO 20 L = 1, MG
      WRITE( ) EM(L), REY(L), ALF(L), TIME(L)
      WRITE( ) (((R (I,J,K,L),I=1,IL(L)),J=1,JL(L)),K=1,KL(L)),
     .          (((RU(I,J,K,L),I=1,IL(L)),J=1,JL(L)),K=1,KL(L)),
     .          (((RV(I,J,K,L),I=1,IL(L)),J=1,JL(L)),K=1,KL(L)),
     .          (((RW(I,J,K,L),I=1,IL(L)),J=1,JL(L)),K=1,KL(L)),
     .          (((RE(I,J,K,L),I=1,IL(L)),J=1,JL(L)),K=1,KL(L))
20    CONTINUE
```

Each of the terms used in the FORTRAN code given above are defined below:

MG	number of grid blocks
IL(L)	maximum grid index in the axial direction for block L
JL(L)	maximum grid index in the radial direction for block L
KL(L)	maximum grid index in the circumferential direction for block L
X(I,J,K,L)	Cartesian coordinate value of x for point (I,J,K) in block L
Y(I,J,K,L)	Cartesian coordinate value of y for point (I,J,K) in block L
Z(I,J,K,L)	Cartesian coordinate value of z for point (I,J,K) in block L

EM(L)	Reference Mach number for block L
REY(L)	Reference Reynolds number for block L
ALF(L)	Reference angle for block L
TIME(L)	Reference time for block L
R (I,J,K,L)	ρ at point (I,J,K) in block L
RU(I,J,K,L)	ρu_x at point (I,J,K) in block L
RV(I,J,K,L)	ρu_y at point (I,J,K) in block L
RW(I,J,K,L)	ρu_z at point (I,J,K) in block L
RE(I,J,K,L)	ρe at point (I,J,K) in block L

The WRITE statements used above can be either unformatted or free format. The equivalent FORTRAN coding using the Scientific DataBase Library (SDBLIB) format [2] which is used in the *ADPAC* codes is given below:

PLOT3D Mesh File Format FORTRAN Coding Example Using SDBLIB

```
CALL QDOPEN( IGRID, FNAME, JE )
CALL QDGETI( IGRID, MG    , JE )
ILENGTH = 3 * MG
CALL QDGEIA( IGRID, IB, ILENGTH, JE )
DO 10 L = 1, MG
IB1 = IB((L-1)*3+1)
IB2 = IB((L-1)*3+2)
IB3 = IB((L-1)*3+3)
ILENGTH = IB1 * IB2 * IB3
CALL QDGEEA( IGRID, X(IPOINT(L)), ILENGTH, JE )
```

```
      CALL QDGEEA( IGRID, Y(IPOINT(L)), ILENGTH, JE )
      CALL QDGEEA( IGRID, Z(IPOINT(L)), ILENGTH, JE )
10    CONTINUE
      CALL QDCLOS( IGRID, JE )
```

PLOT3D Flow File Format FORTRAN Coding Example Using SDBLIB

```
      CALL QDOPEN( IFLOW, FNAME, JE )
      CALL QDGETI( IFLOW, MG   , JE )
      ILENGTH = 3 * MG
      CALL QDGEIA( IFLOW, IB, ILENGTH, JE )
      DO 20 L = 1, MG
      IB1 = IB((L-1)*3+1)
      IB2 = IB((L-1)*3+2)
      IB3 = IB((L-1)*3+3)
      ILENGTH = IB1 * IB2 * IB3
      CALL QDGETE( IFLOW, EM  , JE )
      CALL QDGETE( IFLOW, REY , JE )
      CALL QDGETE( IFLOW, ALF , JE )
      CALL QDGETE( IFLOW, TIME, JE )
      CALL QDGEEA( IFLOW, R (IPOINT(L)), ILENGTH, JE )
      CALL QDGEEA( IFLOW, RU(IPOINT(L)), ILENGTH, JE )
      CALL QDGEEA( IFLOW, RV(IPOINT(L)), ILENGTH, JE )
      CALL QDGEEA( IFLOW, RW(IPOINT(L)), ILENGTH, JE )
```

```
      CALL QDGEEA( IFLOW, RE(IPOINT(L)), ILENGTH, JE )
20    CONTINUE
      CALL QDCLOS( IFLOW, JE )
```

A listing of the additional terms used in the coding above is given below:

QDOPEN	SDBLIB routine to open a file for input or output
QDGETI	SDBLIB routine to get an integer
QDGEIA	SDBLIB routine to get an integer array of length ILENGTH
QDGETE	SDBLIB routine to get a real number
QDGEEA	SDBLIB routine to get a real array of length ILENGTH
QDCLOS	SDBLIB routine to close a file
IGRID	FORTRAN logical unit number for grid input
IFLOW	FORTRAN logical unit number for flow input
JE	An error trigger; 0 for no error, 1 if an error occurs
IB	Integer array containing the IL, JL, and KL grid block sizes
ILENGTH	Integer length of an array of data
IPOINT(L)	Integer pointer for block L to locate the initial memory location for a block of data

In addition, due to the increasing popularity of the *PostScript* page description language, and the variety of devices which can display *PostScript*-based output, a number of plotting procedures included in the *ADPAC* package utilize standard *PostScript*.

3. *CHGRIDV2*: GRID GENERATOR OPERATING INSTRUCTIONS

3.1 Introduction

This chapter contains the computer program User's Manual for the ducted propfan grid generation code *CHGRIDV2*. Grid generation for both unducted and ducted propfan geometries is performed by the single program *CHGRIDV2*. The grid generation source program is written in FORTRAN 77, and has been used successfully on Cray UNICOS and IBM VM/CMS mainframe computer systems, as well as Silicon Graphics 4D workstations using a UNIX operating system.

3.2 Compiling the Source Code

An automated compiling system, or *make* facility is provided to simplify the compilation process for UNIX systems. The source code will compile automatically when the command

make

is issued. Although compiling systems vary from machine to machine, the standard make command will usually suffice to create the object code necessary for final compilation. In order to create the executable code, the specific linking command for a given machine must be used. Since the target systems for this code were intended to be a Silicon Graphics workstation, a Cray computer, or an IBM workstation, separate facilities are provided for each machine. On Silicon Graphics workstation, two options are available. *CHGRIDV2* can support a limited amount of interactive graphics during the grid generation process, or simply run as a one pass grid generation scheme. The interactive feature of the code is invoked during compilation. To create a graphics oriented version of the code, issue the command:

make graphics

The standard one-pass version of the code is assumed by the standard command

make

On the Cray computer, the command:

make cray

will perform the necessary compilation. If compilation on an IBM workstation running the AIX operating system is desired, enter

make aix

Before attempting to run the program, it is necessary to set the maximum array size required for a given analysis prior to the compilation process. Array dimensions in the programs are specified by a PARAMETER statement. The PARAMETER statement is listed in the file **PARAMETER.INC** as:

PARAMETER(IJX=100, KX=31, NX=1)

PARAMETER(IXT=2(NX+1)*IJX)

and appears in every subroutine through an **INCLUDE** statement. Another important parameter, PRECIS, is based on the precision of the host computer and is determined internally.

The individual variables in the PARAMETER statements are defined as:

IJX This variable should approximate the maximum number of points in the axial direction per blade row. Since *CHGRIDV2* can generate grids for multiple blade row geometries, it is often easier to estimate the number of axial grid points per blade row rather than the total number of axial grid points. The actual array storage for the axial grid points is subsequently determined by PARAMETER IXT (below). IJX is also used to determine the maximum number of points in the radial direction.

KX The maximum number of points in the circumferential direction.

NX The maximum number of blade rows (this is explained in more detail in Ref [6]).

IXT The maximum number of points in the axial direction (IXT = 2(NX+1)*IJX). The formula for IXT is somewhat arbitrary and is based on experience.

The values for each parameter listed above are typical for medium-sized grids.

Several rules for storage must be observed for proper operation of the code. Storage is assigned under the assumption that $IJX > KX$. The parameter NX must be greater than or equal to the number of blade rows (this need never be greater than 1 for single rotor applications). For multiple-block C-grid generation, KX must be greater than or equal to $NPBCAB + 1$ ($NPBCAB$ is an input parameter described later). The first step in any grid generation sequence should be to estimate the size of the grid to be generated and to make sure that adequate array storage is provided for the anticipated grid. If, during the course of the grid generation, a PARAMETER value is exceeded, the code will abort execution and print an error message which defines the point where the array limit was reached. The cause of such problems may be due either to insufficient PARAMETER sizes or incorrect input data. For example, specifying a grid ratio which results in an enormous number of grid points can cause such an error, as can an inadequate PARAMETER dimension. It is not always possible for the code to determine a PARAMETER value which will permit a complete grid generation for a given data set once an abort condition has been reached, and it is therefore up to the user to decide how much a PARAMETER value must be increased to allow the grid generation process to run to completion.

An option is provided in *CHGRIDV2* to arbitrarily adjust the 3/4 radius blade setting angle without modifying the blade geometry parameters. This assumes that the pitch change axis is perpendicular to the axis of rotation, and only a solid body rotation can occur (no deflections). This procedure can occasionally produce unexpected results if the blade geometry is not defined beyond the confines of the hub surface, especially if the hub is highly ramped. This problem is aggravated when the blade is highly swept, as it is possible to generate radial running grid lines which have multiple hub intersections. This will result in a failure for the grid generation process and most often results in an error message stating that subroutine INRSCT did not converge. One known cure for this problem is to reduce the overall grid size, especially in the region of high hub curvature, as this will often ease the multiple intersection problem. Unfortunately, for numerical accuracy, this is not normally a desirable action. This problem is being addressed in a future stage of this program.

For plotting results, *CHGRIDV2* produces an output file with data formatted for the *PLOT3D* [3], *SURF* [4], and *FAST* [5] graphics software packages developed at the NASA Ames Research Center. This software may be used to view the grid on a graphics workstation as a means of verifying the grid quality before invoking the flow solver. The grid itself is written in PLOT3D multiple-grid format (see Chapter 2). Additional PLOT3D format data files can be generated for debugging purposes by small changes in the input. Some PostScript output is also available on FORTRAN unit 15, although this is primarily for analyzing the quality of the cowl C-grid in the multiblock grid generation.

The input/output files used by the grid generation codes are described in the first section below. This section is then followed by a more detailed section which describes the function of the individual subroutines in the computer program and a flowchart of the program execution process.

Geometric data may be input in either dimensional, or nondimensional form; however, it is a requirement that all geometric components be described in a consistent manner (either all nondimensional or all dimensional). The final grid coordinates are nondimensionalized by the maximum propfan blade diameter.

3.3 Input/Output File Description

A sample input data file for *CHGRIDV2* is shown in Fig. 3.1. The input file consists of an arbitrary number of header lines which is terminated by a line beginning with a "+" character. Beyond this line, the file follows the structured format which is given in Fig. 3.1. The structured data format is organized in fields of 10 characters, whether integer or real. Integer data must be right justified in their respective fields. Real data can be placed anywhere in their field. A brief description of the variables used in the data file is given below. A number of these input parameters are illustrated graphically in Fig. 3.2.

VARIABLE	DESCRIPTION
TITLE	An 80 character title for the mesh generation.
NBLROW	Number of blade rows: for the current aerodynamic code *AOA*, this must be 1, under other circumstances larger numbers may be desired.
IGEOM	Internal/external geometry parameter: if = 1, internal flow (e.g. compressor or turbine); blade extends from hub to case with no clearance. if = 2, external flow (e.g. propfan, ducted propfan); clearance region exists between blade tip and outer boundary.
ITHETA	Theta input format for blade definitions: if = 1, a mean camber line and tangential thickness are input,

```
COMMENT LINES ON TOP
GRID GENERATION INPUT BEGINS WITH THE FIRST LINE STARTING WITH A "+"

THIS IS A SAMPLE INPUT DATA FILE FOR CHGRIDV2
THIS PROGRAM WAS DEVELOPED UNDER NASA CONTRACT NAS3-25270

+TITLE-------------------------------------------------------------------
 GENERIC DUCTED PROPFAN-DUCTED CLIPPED SR7 GEOMETRY
+NBLROW--++IGEOM---++ITHETA--++IWRITE--++ICOWL---++--------++IORTH---++IDEBUG--+
        1        2        2        2        2                 1        0
+NINPLS--++NEXPLS--++NBLPTZ--++NBLPTR--++NBLPTT--++NCITER--++ISOLVE--++BETAC---+
        1        1       21       15       15        3        0      1.1
+ZINLET--++ZEXIT---++CFACLE--++CFACTE--++DFACLE--++DFACTE--++RATCLU--++EPS-----+
 -1.20000  1.25000     0.02     0.02     1.02     1.02      0.0      1.0
+RATIN---++RATEX---++RATBB---++RATBLZ--++RATBLR--++RATBLT--++RATTOB--++--------+
  1.25000  1.35000  1.20000  1.300000   1.3500  1.35000  1.60000
********************************************************************************
+NHUB----++NOUTB---++--------++--------++--------++--------++--------++--------+
      53        2
+ZHUB,RHUB+--------++--------++--------++--------++--------++--------++--------+
 -80.00000 -20.00000 -0.45000 -0.40000 -0.37000 -0.35000 -0.33600 -0.33000
  -0.32685  -0.32336 -0.31773 -0.31097 -0.30302 -0.29383 -0.28285 -0.25793
  -0.22232  -0.18130 -0.13606 -0.09774 -0.07757 -0.07042 -0.06853 -0.06180
  -0.05049  -0.03837 -0.02810 -0.01930 -0.01181 -0.00522  0.00080  0.00653
   0.01215   0.01808  0.02469  0.03237  0.04148  0.05238  0.06434  0.07312
   0.07788   0.07965  0.09454  0.11764  0.15665  0.21074  0.27769  0.35493
   0.43958   0.58799  0.75179  0.92019  1.09000
   0.00200   0.00200  0.00200  0.00200  0.00260  0.00400  0.00600  0.00800
   0.01000   0.01300  0.02177  0.03076  0.03877  0.04635  0.05436  0.06875
   0.08364   0.09408  0.09971  0.10164  0.10204  0.10239  0.10251  0.10301
   0.10421   0.10611  0.10819  0.11031  0.11257  0.11463  0.11653  0.11837
   0.12065   0.12307  0.12572  0.12874  0.13223  0.13630  0.14061  0.14368
   0.14530   0.14590  0.15081  0.15819  0.16767  0.17458  0.17754  0.17486
   0.16806   0.15661  0.15465  0.15465  0.15465
+ZOUTB,ROUTB-------++--------++--------++--------++--------++--------++--------+
  -10.0000  10.000
   1.7000   1.700
********************************************************************************
+NCOWL---++NCOWLI--++NCOWLO--++NPBCAB--++--------++--------++--------++--------+
      75       11       11        7
+ZCOWL,RCOWL-------++--------++--------++--------++--------++--------++--------+
  0.250000  0.249464  0.247860  0.245194  0.241480  0.236734  0.230970  0.224220
  0.216504  0.207864  0.198340  0.187960  0.176774 -0.164834 -0.176774 -0.187960
 -0.198340 -0.207864 -0.216504 -0.224220 -0.230970 -0.236734 -0.241480 -0.245194
 -0.247860 -0.249464 -0.250000 -0.249464 -0.247860 -0.245194 -0.241480 -0.236734
 -0.230970 -0.224220 -0.216504 -0.207864 -0.198340 -0.187960 -0.176774 -0.164834
 -0.152190 -0.138894 -0.125000 -0.110570 -0.095670 -0.080360 -0.064706 -0.048776
 -0.032630 -0.016350  0.000000  0.016350  0.032630  0.048774  0.064706  0.080360
  0.095670  0.110570  0.125000  0.138894  0.152190  0.164834  0.176774  0.187960
  0.198340  0.207864  0.216504  0.224220  0.230970  0.236734  0.241480  0.245194
  0.247860  0.249464  0.250000
  0.410000  0.410010  0.410040  0.410067  0.410083  0.410075  0.410047  0.410005
  0.409958  0.409910  0.409857  0.409797  0.409725  0.404410  0.404362  0.404385
  0.404480  0.404638  0.404887  0.405247  0.405617  0.406077  0.406807  0.407440
  0.408125  0.408895  0.410000  0.410750  0.411665  0.413025  0.414677  0.416155
  0.417597  0.419175  0.420660  0.422092  0.423482  0.424795  0.426020  0.427153
  0.428175  0.429077  0.429845  0.430467  0.430922  0.431187  0.431250  0.431113
  0.430775  0.430250  0.429552  0.428700  0.427725  0.426660  0.425538  0.424382
  0.423212  0.422045  0.420887  0.419752  0.418655  0.417597  0.416587  0.415635
  0.414738  0.413900  0.413117  0.412395  0.411742  0.411178  0.410720  0.410382
  0.410162  0.410040  0.410000
********************************************************************************
+NBLD----++NBLRCS--++NPPRC---++IPCH----++ZPCA----++THPCA---++BETA34--++--------+
       8        8       21        1      0.0      0.0     60.2
+ZBLA,RBLA,THBLA,TTBLA-------++--------++--------++--------++--------++--------+
  -0.07042  -0.06259 -0.05477 -0.04699 -0.03924 -0.03153 -0.02386 -0.01628
  -0.00878  -0.00130  0.00619  0.01346  0.02074  0.02804  0.03535  0.04269
   0.05004   0.05742  0.06481  0.07222  0.07965
  -0.07840  -0.07040 -0.06241 -0.05443 -0.04648 -0.03857 -0.03070 -0.02287
  -0.01513  -0.00745  0.00021  0.00786  0.01530  0.02277  0.03026  0.03777
   0.04530   0.05285  0.06041  0.06800  0.07560
  -0.08927  -0.08096 -0.07266 -0.06436 -0.05608 -0.04782 -0.03959 -0.03140
  -0.02326  -0.01519 -0.00719  0.00081  0.00875  0.01654  0.02436  0.03219
   0.04005   0.04792  0.05581  0.06372  0.07164
  -0.10222  -0.09346 -0.08470 -0.07594 -0.06718 -0.05844 -0.04971 -0.04101
  -0.03235  -0.02372 -0.01517 -0.00667  0.00183  0.01025  0.01855  0.02690
   0.03526   0.04363  0.05203  0.06043  0.06886
```

Figure 3.1: Sample input data file for grid generation

-0.11092	-0.10175	-0.09257	-0.08339	-0.07421	-0.06504	-0.05588	-0.04674
-0.03762	-0.02853	-0.01948	-0.01050	-0.00152	0.00741	0.01626	0.02510
0.03396	0.04283	0.05172	0.06062	0.06953			
-0.10567	-0.09655	-0.08743	-0.07831	-0.06919	-0.06008	-0.05097	-0.04189
-0.03281	-0.02376	-0.01475	-0.00577	0.00321	0.01214	0.02102	0.02992
0.03883	0.04775	0.05668	0.06562	0.07457			
-0.08982	-0.08126	-0.07270	-0.06414	-0.05558	-0.04704	-0.03850	-0.02997
-0.02145	-0.01295	-0.00447	0.00401	0.01246	0.02089	0.02933	0.03777
0.04622	0.05468	0.06314	0.07160	0.08007			
-0.06358	-0.05613	-0.04869	-0.04125	-0.03381	-0.02638	-0.01895	-0.01153
-0.00411	0.00330	0.01070	0.01810	0.02549	0.03289	0.04029	0.04769
0.05509	0.06249	0.06990	0.07731	0.08472			
0.10239	0.10294	0.10369	0.10469	0.10595	0.10746	0.10915	0.11119
0.11351	0.11588	0.11825	0.12120	0.12414	0.12704	0.12989	0.13269
0.13544	0.13813	0.14078	0.14337	0.14590			
0.11840	0.11876	0.11930	0.12005	0.12104	0.12229	0.12378	0.12550
0.12754	0.12983	0.13216	0.13456	0.13753	0.14041	0.14325	0.14603
0.14877	0.15145	0.15409	0.15666	0.15919			
0.14224	0.14233	0.14263	0.14312	0.14379	0.14471	0.14589	0.14732
0.14897	0.15095	0.15319	0.15546	0.15789	0.16079	0.16361	0.16639
0.16910	0.17176	0.17438	0.17693	0.17943			
0.17874	0.17901	0.17911	0.17927	0.17967	0.18025	0.18105	0.18213
0.18345	0.18499	0.18689	0.18902	0.19118	0.19357	0.19638	0.19906
0.20169	0.20427	0.20679	0.20926	0.21167			
0.22445	0.22475	0.22492	0.22504	0.22525	0.22562	0.22620	0.22701
0.22805	0.22934	0.23084	0.23273	0.23467	0.23678	0.23921	0.24169
0.24409	0.24644	0.24874	0.25098	0.25317			
0.27612	0.27636	0.27645	0.27653	0.27679	0.27718	0.27773	0.27849
0.27945	0.28059	0.28199	0.28357	0.28518	0.28704	0.28914	0.29112
0.29306	0.29497	0.29683	0.29863	0.30040			
0.32968	0.32972	0.32988	0.33011	0.33045	0.33092	0.33153	0.33226
0.33312	0.33418	0.33533	0.33651	0.33789	0.33939	0.34084	0.34226
0.34365	0.34500	0.34632	0.34761	0.34886			
0.40500	0.40516	0.40532	0.40548	0.40564	0.40580	0.40596	0.40612
0.40628	0.40644	0.40660	0.40676	0.40692	0.40708	0.40724	0.40740
0.40756	0.40772	0.40788	0.40804	0.40820			
-0.22623	-0.11448	-0.06580	-0.00593	0.03783	0.07747	0.11208	0.13846
0.15770	0.17116	0.17992	0.18148	0.18524	0.18762	0.18808	0.18672
0.18416	0.18144	0.17914	0.17788	0.17801			
-0.22180	-0.14123	-0.10573	-0.06270	-0.02902	0.00160	0.03009	0.05502
0.07632	0.09372	0.10771	0.11843	0.12448	0.12892	0.13273	0.13719
0.14188	0.14653	0.15096	0.15519	0.15910			
-0.21743	-0.16069	-0.13115	-0.09805	-0.07011	-0.04451	-0.02039	0.00151
0.02138	0.03888	0.05401	0.06720	0.07870	0.08821	0.09698	0.10506
0.11278	0.12067	0.12867	0.13656	0.14430			
-0.22009	-0.17785	-0.15193	-0.12440	-0.09990	-0.07691	-0.05491	-0.03430
-0.01496	0.00299	0.01943	0.03450	0.04855	0.06161	0.07367	0.08531
0.09651	0.10748	0.11820	0.12866	0.13889			
-0.22349	-0.18910	-0.16466	-0.13966	-0.11662	-0.09455	-0.07318	-0.05272
-0.03311	-0.01442	0.00338	0.02025	0.03646	0.05204	0.06701	0.08151
0.09562	0.10941	0.12280	0.13576	0.14805			
-0.19509	-0.16591	-0.14364	-0.12121	-0.09998	-0.07939	-0.05926	-0.03966
-0.02054	-0.00199	0.01598	0.03346	0.05037	0.06683	0.08275	0.09822
0.11322	0.12781	0.14191	0.15547	0.16812			
-0.15033	-0.12553	-0.10575	-0.08594	-0.06705	-0.04864	-0.03057	-0.01289
0.00442	0.02140	0.03806	0.05435	0.07034	0.08590	0.10119	0.11613
0.13073	0.14494	0.15867	0.17184	0.18402			
-0.09282	-0.07252	-0.05588	-0.03926	-0.02329	-0.00766	0.00771	0.02279
0.03766	0.05229	0.06671	0.08089	0.09482	0.10855	0.12203	0.13526
0.14816	0.16076	0.17300	0.18478	0.19570			
-0.22623	-0.25766	-0.25881	-0.25715	-0.25046	-0.24091	-0.22859	-0.21185
-0.19041	-0.16474	-0.13510	-0.10277	-0.07418	-0.04316	-0.01028	0.02425
0.05918	0.09269	0.12376	0.15196	0.17801			
-0.22180	-0.23798	-0.22938	-0.22032	-0.20653	-0.19103	-0.17463	-0.15694
-0.13762	-0.11651	-0.09377	-0.06880	-0.04048	-0.01104	0.01729	0.04308
0.06803	0.09214	0.11529	0.13745	0.15910			
-0.21743	-0.22212	-0.20847	-0.19620	-0.18074	-0.16395	-0.14652	-0.12818
-0.10906	-0.08898	-0.06796	-0.04625	-0.02392	-0.00143	0.02083	0.04278
0.06427	0.08517	0.10544	0.12510	0.14430			
-0.22009	-0.21557	-0.19920	-0.18427	-0.16738	-0.14978	-0.13197	-0.11370
-0.09505	-0.07598	-0.05637	-0.03634	-0.01600	0.00444	0.02471	0.04480
0.06456	0.08392	0.10277	0.12103	0.13890			
-0.22349	-0.21227	-0.19376	-0.17620	-0.15761	-0.13876	-0.11988	-0.10086
-0.08176	-0.06255	-0.04321	-0.02382	-0.00435	0.01505	0.03434	0.05353
0.07264	0.09166	0.11059	0.12936	0.14805			
-0.19509	-0.18096	-0.16248	-0.14470	-0.12627	-0.10775	-0.08929	-0.07077
-0.05226	-0.03375	-0.01518	0.00331	0.02187	0.04033	0.05875	0.07715
0.09554	0.11384	0.13209	0.15024	0.16813			
-0.15032	-0.13572	-0.11844	-0.10164	-0.08457	-0.06750	-0.05050	-0.03353
-0.01658	0.00034	0.01715	0.03399	0.05071	0.06742	0.08415	0.10087
0.11762	0.13434	0.15109	0.16771	0.18402			
-0.09282	-0.07935	-0.06428	-0.04962	-0.03484	-0.02008	-0.00543	0.00917
0.02374	0.03826	0.05271	0.06713	0.08149	0.09586	0.11021	0.12457
0.13894	0.15326	0.16758	0.18182	0.19570			

if = 2, surface theta coordinates are input (preferred).

IWRITE Format of mesh (not used).

ICOWL Cowl geometry parameter:

if = 0 no cowl, no cowl geometry is read,

if = 1 generate an H-grid for the ducted geometry (cowl geometry is read); this is suitable for use with the *HPRO3D* code developed under Task I of this contract.

if = 2, generate a multiple-block C-grid for the ducted geometry (cowl geometry is read).

IORTH C-grid orthogonality control parameter:

if = 0, cowl surface points for the C-grid are clustered based on arc length interpolation,

if = 1, cowl surface points for the C-grid are clustered based on an algorithm which attempts to maintain orthogonality.

IDEBUG Debugging parameter:

if = 0 no extra debugging plots,

if = 1 several extra *PLOT3D* data sets are produced for debugging.

NINPLS Determines the number of constant spacing inlet planes added to the grid inlet (NINPLS-1 planes are added, see Fig. 3.2). Typically, this variable is given a value of 1, which ensures that the inlet plane of the grid has an axial location equal to ZINLET. Larger values may be added to move the inlet plane farther upstream.

NEXPLS Determines the number of constant spacing exit planes added to the grid exit (NEXPLS-1 planes are added, see Fig. 3.2). Typically, this variable is given a value of 1, which ensures that the exit plane of the grid has an axial location equal to ZEXIT. Larger values may be added to move the exit plane farther downstream.

NBLPTZ Number of points on the airfoil in the axial direction (see Fig. 3.2). This must be an odd number. If an even number is entered, the program will issue an error message and abort. For a reasonably accurate solution, this should probably be given a value of 21 or larger, although smaller values may be used for testing purposes.

NBLPTR Number of points on the airfoil in the radial direction (see Fig. 3.2). This must be an odd number. If an even number is entered, the program will issue an error message and abort. For a reasonably accurate solution, this should probably be given a value of 15 or larger (much larger for viscous flow calculations), although smaller values may be used for testing purposes.

NBLPTT Number of points between airfoils in the circumferential direction. This must be an odd number $\leq$ PARAMETER KX. If an even number is entered, the program will issue an error message and abort. For a reasonably accurate solution, this should probably be given a value of 15 or larger (much larger for viscous flow calculations), although smaller values may be used for testing purposes.

NCITER Number of iterations for the C-grid solver.

if = 0, the C-grid is generated using the algebraic solver only. Any value larger than 0 dictates the number of iterations for the C-grid elliptic grid generation scheme which uses the algebraic solution as an initial guess. It is usually a good idea to permit a small number iterations (3) of the elliptic solver to smooth the coordinates.

ISOLVE C-grid generation algorithm control parameter:

if = 0, C-grid is generated using the standard elliptic solver,

if = 1, C-grid is generated using the variational solver. At this point, 0 is the only recommended value.

BETAC C-grid generation clustering parameter;

this must be greater than 1.0, more clustering near the cowl surface as BETAC approaches 1.0. A reasonable starting point for this variable is 1.5 (1.05 for a viscous flow calculation).

ZINLET Axial location of the initial inlet boundary plane (before constant spacing inlet planes are added, see Fig. 3.2).

ZEXIT Axial location of the initial exit boundary plane (before constant spacing exit planes are added, see Fig. 3.2).

CFACLE Factor which determines the axial extent of the C-grid upstream of the cowl leading edge (multiplied by cowl axial chord, see Fig. 3.2). Normally, this value should be roughly equivalent to the leading edge thickness based on fraction of chord. A value of 0.1 is generally acceptable for most geometries.

CFACTE Factor which determines the axial extent of the C-grid downstream of the cowl trailing edge (multiplied by cowl axial chord, see Fig. 3.2).

Normally, this value should be roughly equivalent to the trailing edge thickness based on fraction of chord. A value of 0.1 is generally acceptable for most geometries.

DFACLE Factor which determines the radial width of the C-grid upstream of the cowl leading edge. The radial width is determined by DFACLE * (blade leading edge tip gap). If DFACLE ≥ 1, then the radial width is larger than the tip gap at the blade leading edge; if DFACLE ≤ 1, then the radial width is smaller than the tip gap at the blade leading edge (see Fig. 3.2). Reasonable values for this variable are highly geometry dependent. A value of 2.0 is recommended for initial attempts at most geometries.

DFACTE Factor which determines the radial width of the C-grid downstream of the cowl trailing edge. The radial width is determined by DFACTE * (blade trailing edge tip gap). If DFACTE ≥ 1, then the radial width is larger than the tip gap at the blade trailing edge; if DFACTE ≤ 1, then the radial width is smaller than the tip gap at the blade trailing edge (see Fig. 3.2). Reasonable values for this variable are highly geometry dependent. A value of 2.0 is recommended for initial attempts at most geometries.

RATCLU Factor which determines the percentage of importance of curvature versus grid aspect ratio used to determine the distribution of grid points along the hub (0.0-1.0): if =0.0, no curvature clustering on the hub. if =1.0, hub distribution is determined completely by the curvature of the hub. A reasonable starting value for this variable is 0.25

EPS Implicit smoothing factor for hub curvature distribution. (Used to smooth out irregular curvature data, more smoothing as **EPS** gets larger.) Acceptable values for this parameter can be highly geometry dependent. A reasonable initial value is 2.0.

The following 7 variables control the mesh cell expansion ratios for various regions of the grid. In general, reasonable values for these variables are $1.0 \leq \text{RAT}xxx \leq 1.4$ because of the numerical error with larger ratios. For viscous flow calculations, however, mesh clustering becomes much more important, and the upper limit may be raised to 2.0 to achieve the proper cell spacing near solid surfaces required to adequately resolve viscous shear layers.

RATIN Maximum axial adjacent cell spacing ratio in the inlet region (must be > 1, more clustering as value increases).

RATEX Maximum axial adjacent cell spacing ratio in the exit region (must be > 1, more clustering as value increases).

RATBB Maximum axial adjacent cell spacing ratio between blade rows (must be > 1, more clustering as value increases).

RATBLZ Maximum axial adjacent cell spacing ratio on blades (must be ≥ 1, more clustering as value increases).

RATBLR Maximum radial adjacent cell spacing ratio blades (must be ≥ 1, more clustering as value increases).

RATBLT Maximum circumferential adjacent cell spacing ratio blade to blade (must be ≥ 1, more clustering as value increases).

RATTOB Maximum radial adjacent cell spacing ratio from tip (unducted H-grids) or the cowl upper surface (ducted H-grids) or the C-grid upper boundary (ducted C-grids) to the outer boundary (must be > 1, more clustering as value increases).

NHUB Number of (z,r) hub coordinate input pairs to be read.

NOUTB Number of (z,r) outer boundary coordinate pairs to be read.

ZHUB,RHUB Hub coordinates. All of the hub axial coordinates are given first (8 per line), followed separately by the radial coordinates. The number of points input is determined by **NHUB**.

ZOUTB,ROUTB Outer boundary coordinates. All of the outer axial coordinates are given first (8 per line, followed separately by the radial coordinates. The number of points input is determined by **NOUTB**.

The following 5 variables are used only for a ducted geometry. They should only be included for a ducted case.

NCOWL Number of (z,r) cowl coordinate input pairs to be read.

NCOWLI Number of axial grid lines between cowl leading edge and first blade row leading edge. This must be an odd number. If an even number is entered, the program will issue an error message and abort. For most cases, a value of 11 or above is appropriate.

NCOWLO Number of axial grid lines between last blade row trailing edge and cowl trailing edge. This must be an odd number. If an even number is entered, the program will issue an error message and abort. For most cases, a value of 11 or above is appropriate.

NPBCAB Number of radial grid lines between cowl lower surface and blade tip (number of grid lines in blade tip clearance region). For multiple-block C-grids, this defines the number of grid points in the radial direction for the C-grid. This variable will normally have a value greater than 3 and less than 21. Larger values will generate grid cells with extremely high aspect ratios in the clearance flow region. This value must be less than or equal to KX-1.

ZCOWL,RCOWL Cowl coordinates. The coordinates are input in a wraparound fashion beginning at the trailing edge and proceeding in the clockwise direction. The trailing edge point must be duplicated as the last coordinate. All of the cowl axial coordinates are given first (8 per line), followed separately by the radial coordinates. The number of points input is determined by **NCOWL**.

The remaining variables define the propfan blades themselves. The blades are defined by roughly radial slices with identical axial and radial coordinates on each side of the airfoil. The radial cross section description is illustrated in Fig. 3.3. The airfoil coordinate description along each cross section is illustrated in Fig. 3.4. The radial slices are given in order from the hub to the tip. The tip cross section must be specified in a manner which provides some clearance between the tip of the blade and the cowl lower surface, even if the two parts are actually mated.

It should be mentioned that the blade thickness becomes zero at the grid line representing the blade tip. This feature implies some inconsistency in the representation of the true geometry, but ensures that the numerical scheme maintains a conservative property. This implies a small source of error due to the H-grid representation of the blade and also causes the cells in this area to become distorted, which may adversely affect the accuray of the solution.

The 8 variables which follow are duplicated for each blade row when more than one row is considered.

NBLD Number of blades in this blade row.

NBLRCS Number of radial cross sections used to define the blade (see Fig. 3.3).

NPPRC Number of axial points per radial cross section used to define the blade (see Fig. 3.4).

IPCH Pitch change trigger: if = 0 no pitch change is performed, if = 1 the pitch change variables **ZPCA**, **THPCA**, and **BETA34** are read in and the blade setting angle is adjusted to the value specified by **BETA34** (see Fig. 3.5).

ZPCA Axial location of the blade pitch change axis. The pitch change axis is assumed perpendicular to the axis of rotation (see Fig. 3.5).

THPCA Circumferential position of the blade pitch change axis. The pitch change axis is assumed perpendicular to the axis of rotation.

BETA34 Absolute 3/4 radius blade setting angle. This angle is measured positive from the plane of rotation. The blade setting angle reference plane is the same for either positive or negative rotation (see Fig. 3.5).

ZBLA,RBLA, Axial and radial blade coordinates.

THBLA,TTBLA Circumferential blade coordinates.

Each of the blade axial coordinates is given first (8 per line) running chordwise from leading edge to trailing edge for each cross section, (cross sections are given from hub to tip) followed by all the radial coordinates. The blade circumferential coordinates are given next, and may be in two different formats depending on the variable **ITHETA**. The preferred format is **ITHETA**=2, where the actual tangential surface coordinates are given ordered by increasing theta. For a blade row rotating in the negative direction, the surfaces would be ordered as pressure then suction surface (see Fig. 3.4).

3.4 File Names

All standard files for the grid generation process follow a consistent naming convention based on the UNIX filename structure. The key to this naming process is the *case* name. All filenames have the form:

case.extension

where *case* is a unique name specified by the user to describe the geometry or flow condition being investigated, and *extension* describes the type of file. The *case* name should be available to the code in a file named:

case.def

where **case.def** is a one line file containing the *case* name.

The grid generation system produces two primary types of output files: the standard output, and the grid or grids, depending on whether a C-grid or H-grid is

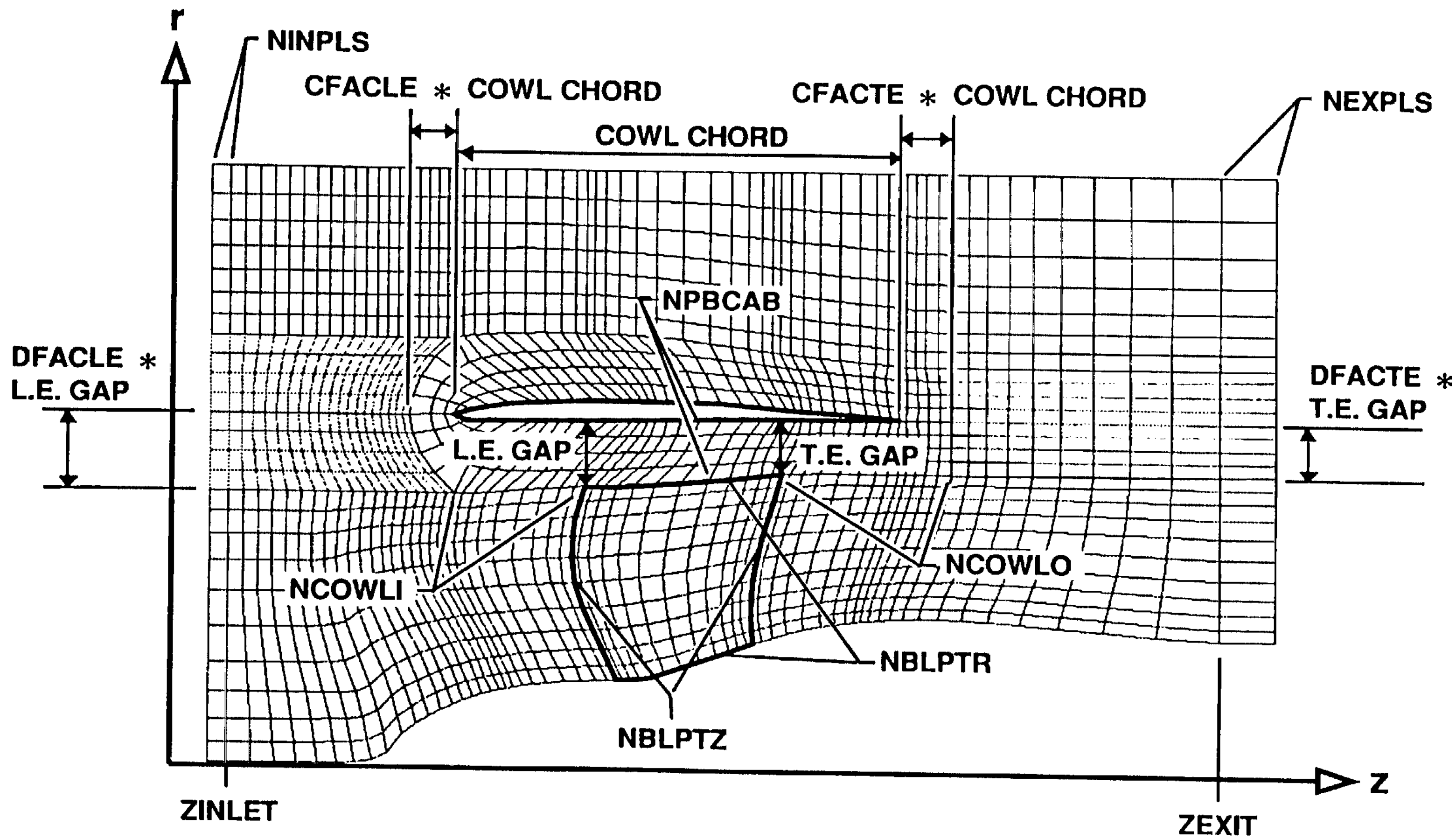

Figure 3.2: Cowl C-grid grid generation parameter description

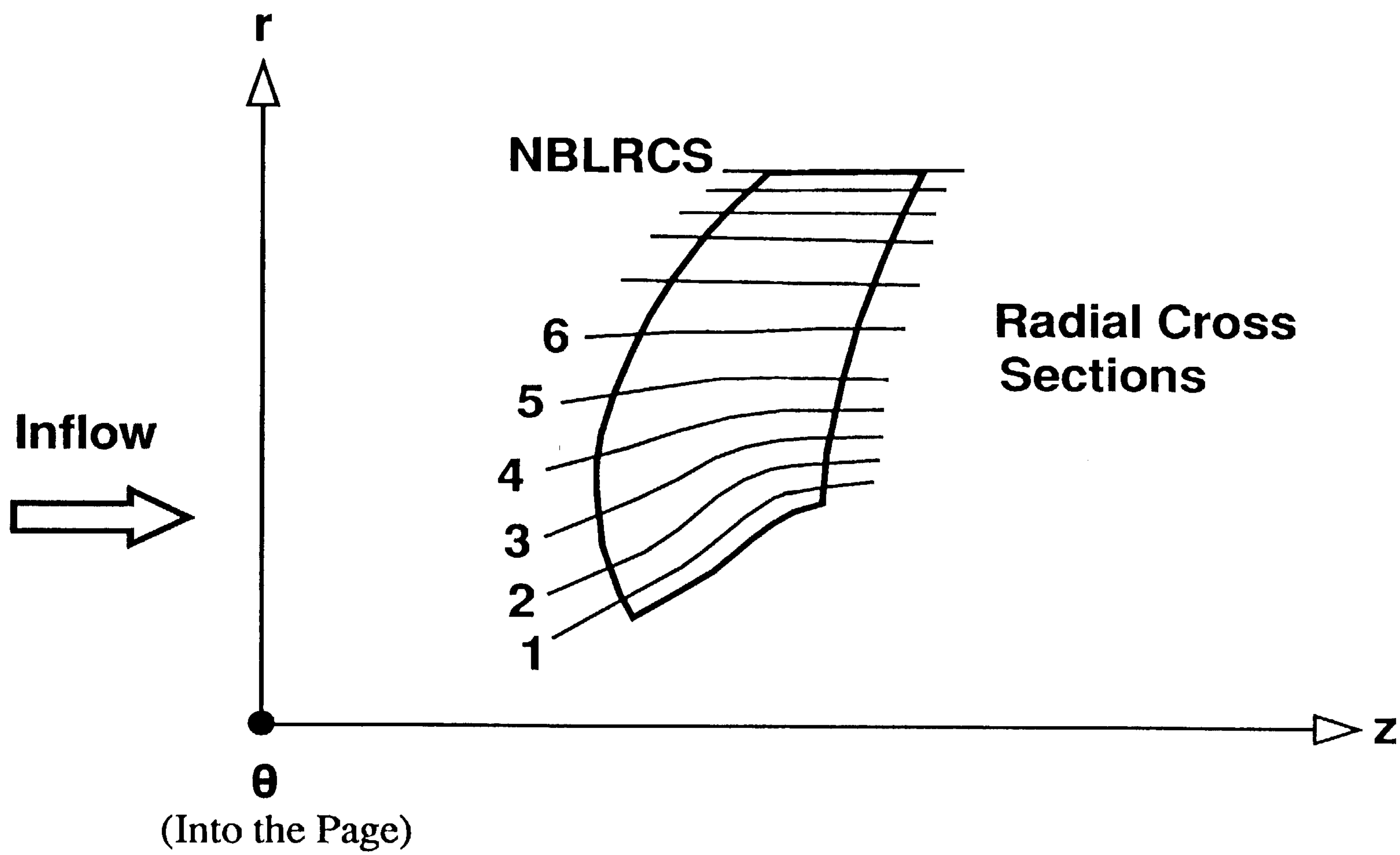

Figure 3.3: Description of blade radial cross section data.

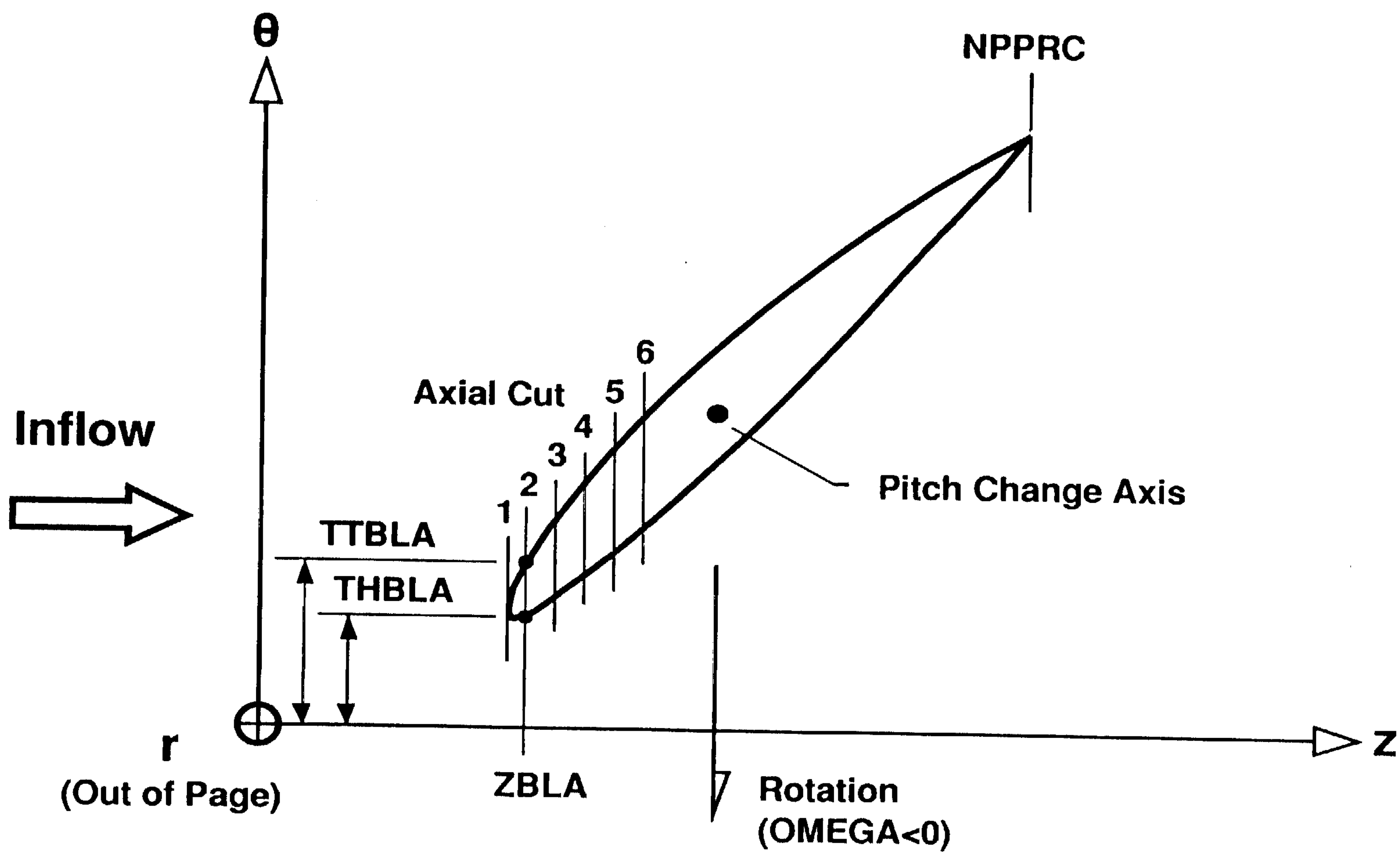

Figure 3.4: Description of blade section geometry data.

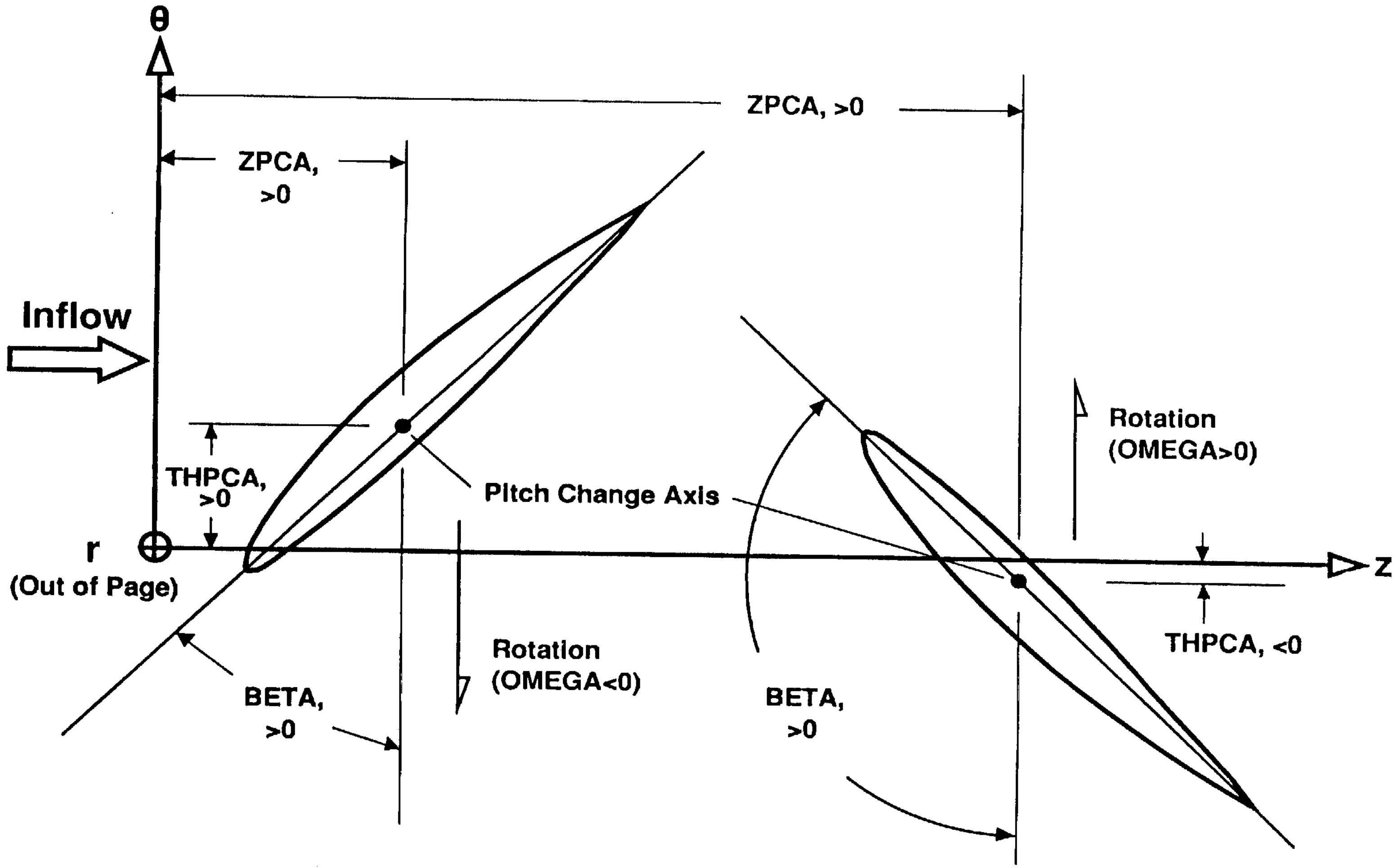

Figure 3.5: Description of blade setting angle and blade location parameters.

Table 3.1: Description of input/output files and UNIX-based filenames for *CHGRIDV2* grid generation program

Name	Description
case.def	File containing the *case* name
case.ggenin	Grid generation input file
case.ggenout	Grid generation output file
case.mesh	Output mesh file (*PLOT3D* compatible)
case.ggenin.new	Updated grid generation input file from interactive run
fort.15	*PostScript* carpet plots of C-grid mesh parameters
fort.16	Partial mesh file (debug runs only)
fort.17	Partial mesh file (debug runs only)
fort.27	Partial mesh file (debug runs only)
fort.37	Partial mesh file (debug runs only)
fort.45	Partial mesh file (debug runs only)
fort.57	Partial mesh file (debug runs only)
fort.67	Partial mesh file (debug runs only)
fort.71	Partial mesh file (debug runs only)
fort.72	Partial mesh file (debug runs only)
fort.87	Partial mesh file (debug runs only)
fort.97	Partial mesh file (debug runs only)

being produced. A description of these files is given in Table 3.1.

A typical H-grid (NCOWL=1) and a multiple-block C-grid (NCOWL=2) generated from the sample data set given in Fig. 3.1 for a ducted propfan are shown in Fig. 3.6. The blade tip gap has been expanded to illustrate the grid in this region.

The *PostScript* plots given in the file *fort.15* are intended to give the user a rapid picture of the C-grid mesh quality in terms of the variation of grid coordinate derivatives, measure of orthogonality, etc. This should not be considered the only means of evaluating mesh quality, but is intended to be used as a tool during the mesh evaluation process.

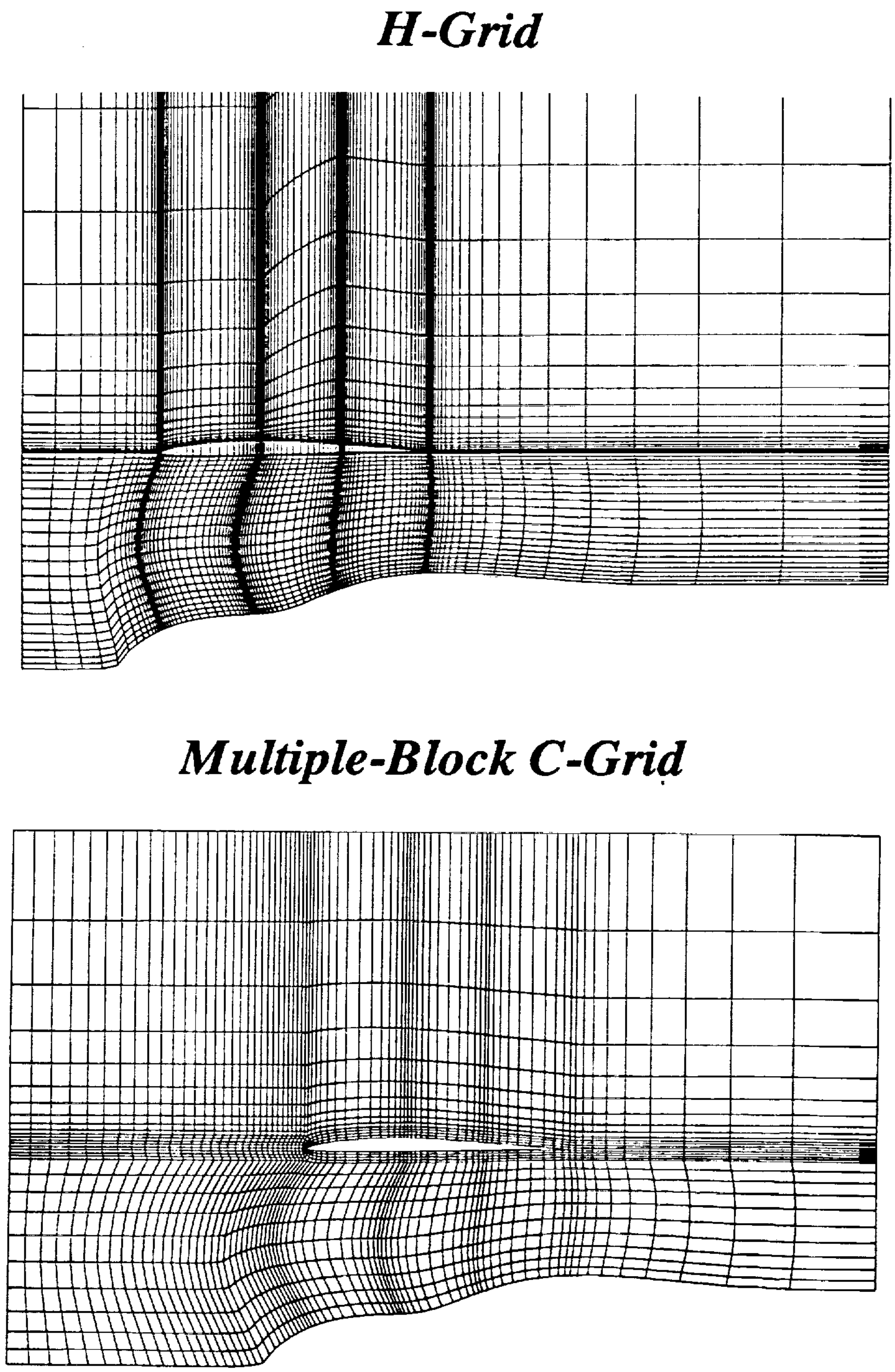

Figure 3.6: Sample grids for a ducted propfan

The partial mesh files given in files *fort.16-fort.73* allow the user to track the mesh generation process. If the code is unable to complete the mesh, the region where the failure occurred can be identified. Each file pertains to a specific aspect of the mesh generation process, which may be identified by the descriptions below:

fort.16	Blade inlet region meridional plane grid
fort.17	Meridional plane of grid below blade tip
fort.27	Meridional plane of grid below blade tip after cowl inlet modifications
fort.37	Meridional plane of grid below blade tip after cowl inlet modifications and reinterpolation of radial rays
fort.45	Meridional plane of cowl C-grid
fort.57	Meridional plane of cowl C-grid outer boundaries only
fort.67	Meridional plane of grids forming blocks 1, 2, and 4
fort.71	Meridional plane of grid below tip and downstream of blade
fort.72	Meridional plane of blade grid
fort.87	Meridional plane of modified cowl C-grid
fort.97	Meridional plane of grids forming blocks 1, 2, 4, and 5

These partial mesh files are written as two-dimensional *PLOT3D* unformatted files and are therfore machine-dependent.

The input and output files are in standard ASCII format. The mesh file is stored in a machine independent format [2], and is compatible with the *PLOT3D* multiple-grid, binary file description (see Chapter 2). The standard input and standard output files are automatically redirected at runtime so execution is initiated simply as:

3.5 Subroutine Description

A list of the grid generation program subroutines and their functions is given below for reference. A skeleton program flowchart is illustrated in Fig. 3.7.

SUBROUTINE	DESCRIPTION
CHGRID	Main calling routine. (Separate graphics and non-graphics versions are included with the source code distribution.)
ALPHM	Plotting routine - specifically for alphanumeric labels, etc.
ANALYZ	Carpet plotting routine - specifically for C-grids.
CCOWL	Routine for setting up and constructing C-grid boundaries.
CGRID	Routine for determining C-grid interior point distribution.
CLUSTR	Routine for determining grid cluster along hub.
CONVAS	Routine to convert array storage - used in conjunction with the SD-BLIB input/output routines.
COWEX	Routine which determines the outer boundary of the exit region.
COWIN	Routine which determines the outer boundary of the inlet region.
CREFIN	Routine for determining grid cluster for cowl C-grid.
CURPLT	Plotting routine - specifically for plotting data as curves.
DIST	Routine used in conjunction with CLUSTR to distribute grid points along the hub boundary.
DOMENU	Interactive grid generation menu routine for IRIS graphics.

ADPAC-CHGRIDV2 Program Calling Tree

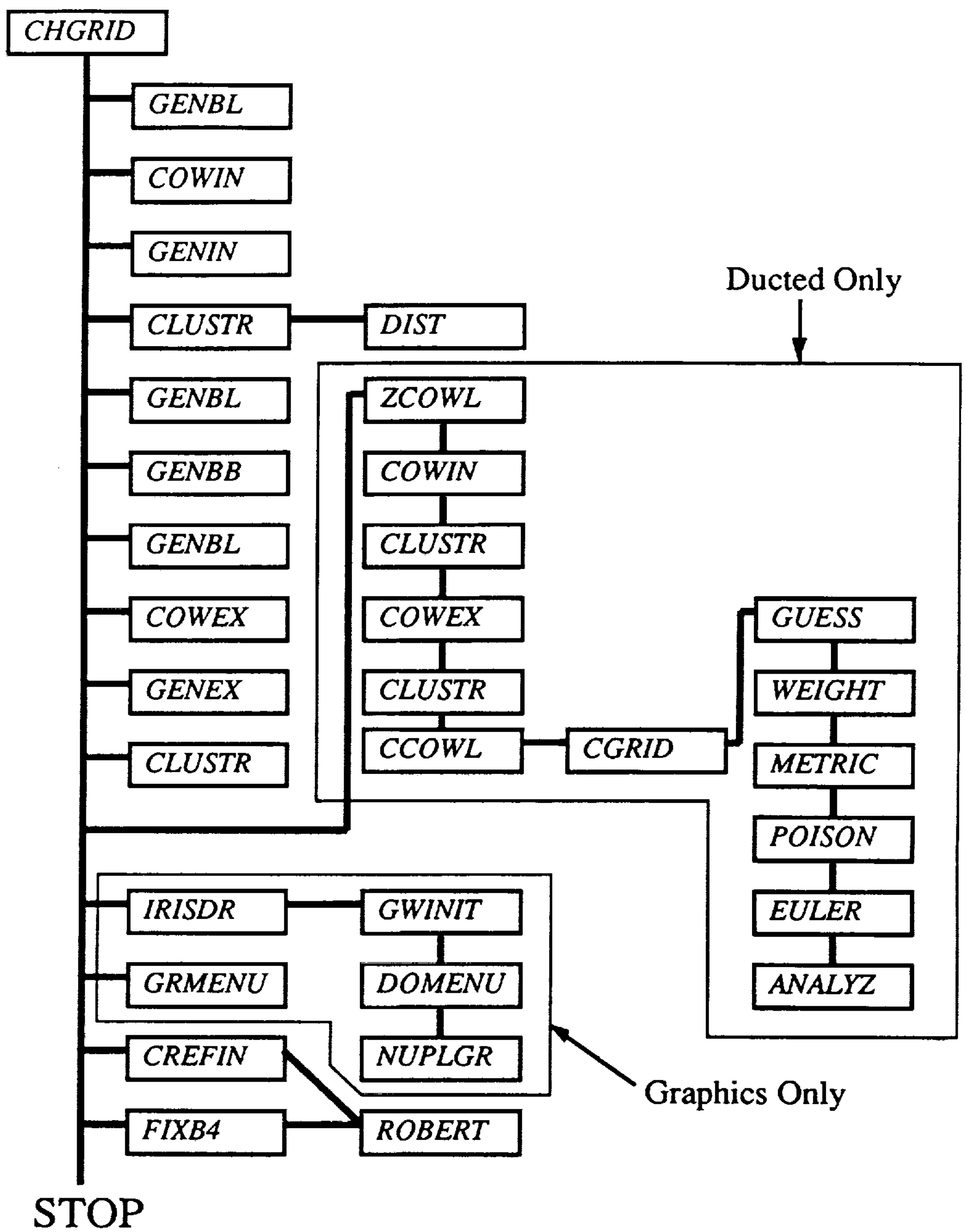

Figure 3.7: Program flowchart for ducted propfan grid generation

END Plotting routine - specifically for ending a plot.

ERROR Routine for internal evaluation of potential input data or grid errors.

EULER Variational-based interior point solver for C-grids.

FIXB4 Routine to adjust block # 4 after C-grid clustering.

GENBB Routine for generating grid point distribution between blades.

GENBL Routine for determining blade grid point distribution.

GENEX Routine for determining exit region grid point distribution.

GENIN Routine for determining inlet region grid point distribution.

GRMENU Interactive graphics grid generation menu routine.

GUESS Routine for generating an initial guess for C-grid interior point grid generation scheme.

GWINIT Graphics window initialization routine.

INRSCT Spline-based routine for determining the intersection of two curves.

INTCHA Integer to character conversion routine.

INTERP Polynomial-based interpolation routine.

INTERP Lower order interpolation routine.

IRISDR IRIS graphics routine.

LTRIM Routine to determine the length of a trimmed character string.

METRIC Routine for determining metric terms in C-grid generation.

NUMPTS Routine for determining the number of grid lines required for a specified grid cell ratio in a given region.

NUPLGR Graphics routine for plotting a grid.

POISON Poisson-based interior point solver for C-grid.

PRINT0 Output routine for grid generation.

ROBERT Exponential-based grid clustering routine.

SCALP Plotting routine - specifically for generating scales.

SPLINT Cubic spline polynomial interpolation routines.

START Plotting routine - specifically for initiating a plot.

TRIM Routine to trim trailing blanks from a character string.

WEIGHT Grid point weighting routine for variational grid generation.

ZCOWL Routine for generating cowl grids.

It should also be mentioned that a number of routines from the SDBLIB library are also included in the source code distribution, but are not defined in detail here.

3.6 Interactive Graphics Operation

CHGRIDV2 supports a limited amount of graphical interaction when compiled with the

make graphics

option. During the course of the grid generation process, the user will be prompted to open a graphics window with the mouse on an IRIS workstation. This graphics window will then display the current meridional grid. The user can translate or zoom the grid to examine local regions of the grid by pressing the center (zoom) or left (translate) mouse buttons while input is directed to the graphics window, and then dragging the mouse to perform the desired transformation (this operation is similar to mouse operations in the *PLOT3D* code). Once the viewing process is complete, the user is prompted to continue with an interactive menu parameter update procedure. The user may select which input parameter is to be changed, and a new value can be entered. Once the grid parameters have been adjusted, the user can select an option which regenerates the grid, and starts the interactive cycle over again. Once a satisfactory grid has been generated, the interactive process may be discontinued by selecting the appropriate option, and a grid and a restart file (*case*.ggenin.new) are output. The restart file may be used as an input file to regenerate the final grid without going through the interactive process.

3.7 Error Messages

CHGRIDV2 has an extended internal error checking facility which is intended to warn the user of potential problems during the course of a calculation. This section describes the meaning of the error and warning messages produced by *CHGRIDV2* and possible courses of action to correct the errors.

Message: **ERROR DETECTED IN CHGRID: FILE COULD NOT BE OPENED**

Meaning: An error has occurred in subroutine CHGRID while attempting to open a file for input. Check that the file name and path are correct, as well as having read permission on the file.

Message: **CONVAS: ERROR - CANNOT DETERMINE CONVER-**

SION PROCESS

Meaning: An error has occurred in subroutine CONVAS. During the course of a conversion from one array structure to another, CONVAS has discovered that the input and output array sizes are inconsistently specified. This error should never occur.

Message: **INPUT FILE ERROR: NBLROW < 1 OR > NX**

Meaning: An error has been detected in subroutine ERROR. The input variable NBLROW must be greater than 1, but less than program PARAMETER NX. The input variable must be changed, or the code must be recompiled with a larger value of PARAMETER NX.

Message: **INPUT FILE ERROR: IGEOM < 1 OR > 2**

Meaning: An error has been detected in subroutine ERROR. The input variable IGEOM must be either 1 or 2.

Message: **INPUT FILE ERROR: ITHETA < 1 OR > 2**

Meaning: An error has been detected in subroutine ERROR. The input variable ITHETA must be either 1 or 2.

Message: **INPUT FILE ERROR: NINPLS < 1 OR > IJX**

Meaning: An error has been detected in subroutine ERROR. The input variable NINPLS must be greater than 1, but less than program PARAMETER IJX. The input value must be changed, or the code must be recompiled with a larger value of IJX.

Message: **INPUT FILE ERROR: NINPLS > IJX-1-3*NBLROW -**

PROGRAM ABORTED

Meaning: An error has been detected in subroutine ERROR. The value of input variable NINPLS has been found to exceed a global array size constraint for unducted propfans (see Table 3.2). The input value must be changed, or the code must be recompiled with a larger value of IJX.

Message: **INPUT FILE ERROR: NINPLS > IJX-1-1-1-3*NBLROW-PROGRAM ABORTED**

Meaning: An error has been detected in subroutine ERROR. The value of input variable NINPLS has been found to exceed a global array size constraint for ducted propfans (see Table 3.2). The input value must be changed, or the code must be recompiled with a larger value of IJX.

Message: **INPUT FILE ERROR: NEXPLS > IJX-NINPLS-3*NBLROW - PROGRAM ABORTED**

Meaning: An error has been detected in subroutine ERROR. The value of input variable NEXPLS has been found to exceed a global array size constraint for unducted propfans (see Table 3.2). The input value must be changed, or the code must be recompiled with a larger value of IJX.

Message: **INPUT FILE ERROR: NEXPLS > IJX-NINPLS-1-1-3*NBLROW - PROGRAM ABORTED**

Meaning: An error has been detected in subroutine ERROR. The value of input variable NEXPLS has been found to exceed a global array size constraint for ducted propfans (see Table 3.2). The input value must be changed, or the code must be recompiled with a larger value of IJX.

Message: **INPUT FILE ERROR: NBLPTZ > (IJX-NINPLS-NEXPLS)/NBLROW PROGRAM ABORTED**

Meaning: An error has been detected in subroutine ERROR. The value of input variable NBLPTZ has been found to exceed a global array size constraint (see Table 3.2). The input value must be changed, or the code must be recompiled with a larger value of IJX.

Message: **INPUT FILE ERROR: NBLPTZ > (IJX-NINPLS-NEXPLS-1-1)/NBLROW) - PROGRAM ABORTED**

Meaning: An error has been detected in subroutine ERROR. The value of input variable NBLPTZ has been found to exceed a global array size constraint (see Table 3.2). The input value must be changed, or the code must be recompiled with a larger value of IJX.

Message: **INPUT FILE ERROR: NCOWLI > (IJX-NINPLS-NEXPLS-1-NBLPTZ*NBLROW) - PROGRAM ABORTED**

Meaning: An error has been detected in subroutine ERROR. The value of input variable NCOWLI has been found to exceed a global array size constraint (see Table 3.2). The input value must be changed, or the code must be recompiled with a larger value of IJX.

Message: **INPUT FILE ERROR: NCOWLO > (IJX-NINPLS-NEXPLS-**

NCOWLI-NBLPTZ*NBLROW) - PROGRAM ABORTED

Meaning: An error has been detected in subroutine ERROR. The value of input variable NCOWLO has been found to exceed a global array size constraint (see Table 3.2). The input value must be changed, or the code must be recompiled with a larger value of IJX.

Message: **INPUT FILE ERROR: NEXPLS < 1 OR > IJX**

Meaning: An error has been detected in subroutine ERROR. The input variable NEXPLS must be greater than 1, but less than program PARAMETER IJX. The input value must be changed, or the code must be recompiled with a larger value of IJX.

Message: **INPUT FILE ERROR: NBLPTZ < 3 OR > IJX**

Meaning: An error has been detected in subroutine ERROR. The input variable NBLPTZ must be an odd number greater than 3, but less than program PARAMETER IJX. The input value must be changed, or the code must be recompiled with a larger value of IJX.

Message: **INPUT FILE ERROR: NBLPTZ IS NOT AN ODD INTEGER**

Meaning: An error has been detected in subroutine ERROR. The input variable NBLPTZ must be an odd number.

Message: **INPUT FILE ERROR: NBLPTR < 3 OR > IJX**

Meaning: An error has been detected in subroutine ERROR. The input variable NBLPTR must be an odd number greater than 3, but less than program PARAMETER IJX. The input value must be changed, or the code must be recompiled with a larger value of IJX.

Message: **INPUT FILE ERROR: NBLPTR IS NOT AN ODD INTEGER**

Meaning: An error has been detected in subroutine ERROR. The input variable NBLPTR must be an odd number.

Message: **INPUT FILE ERROR: NBLPTT < 3 OR > KX**

Meaning: An error has been detected in subroutine ERROR. The input variable NBLPTT must be an odd number greater than 3, but less than program PARAMETER IJX. The input value must be changed, or the code must be recompiled with a larger value of IJX.

Message: **INPUT FILE ERROR: NBLPTT IS NOT AN ODD INTEGER**

Meaning: An error has been detected in subroutine ERROR. The input variable NBLPTT must be an odd number.

Message: **INPUT FILE ERROR: NCOWLI IS NOT AN ODD INTEGER**

Meaning: An error has been detected in subroutine ERROR. The input variable NCOWLI must be an odd number.

Message: **INPUT FILE ERROR: NCOWLO IS NOT AN ODD INTEGER**

Meaning: An error has been detected in subroutine ERROR. The input variable NCOWLO must be an odd number.

Message: **INPUT FILE ERROR: ICOWL IS NOT = 0, 1, OR 2**

Meaning: An error has been detected in subroutine ERROR. The value of input variable ICOWL must be either 0, 1, or 2.

Message: **INPUT FILE ERROR: NPBCAB+1 > KX**

Meaning: An error has been detected in subroutine ERROR. The value of input variable NPBCAB must be less than PARAMETER KX-1. The input variable must be changed, or the code must be recompiled with a larger value of KX.

Message: **INPUT FILE ERROR: DFACLE <= 0.0**

Meaning: An error has been detected in subroutine ERROR. The value of input variable DFACLE must be greater than 0.0.

Message: **INPUT FILE ERROR: DFACTE <= 0.0**

Meaning: An error has been detected in subroutine ERROR. The value of input variable DFACTE must be greater than 0.0.

Message: **INPUT FILE ERROR: CFACLE <= 0.0**

Meaning: An error has been detected in subroutine ERROR. The value of input variable CFACLE must be greater than 0.0.

Message: **INPUT FILE ERROR: CFACTE <= 0.0**

Meaning: An error has been detected in subroutine ERROR. The value of input variable CFACTE must be greater than 0.0.

Message: **INPUT FILE ERROR: NPBCAB < 2**

Meaning: An error has been detected in subroutine ERROR. The value of input variable NPBCAB must be greater than 2.

Message: **INPUT FILE ERROR: IGEOM MUST = 2 FOR ICOWL.NE.0**

Meaning: An error has been detected in subroutine ERROR. The value of input variable IGEOM must be 2 when ICOWL is non-zero.

Message: **INPUT FILE ERROR: ZCOWL(1), RCOWL(1) MUST = ZCOWL(NCOWL), RCOWL(NCOWL)**

Meaning: An error has been detected in subroutine ERROR. The first and last points of the cowl geometry definition must be identical.

Message: **INPUT FILE ERROR: RAT*xxx* < OR = 1.0**

Meaning: An error has been detected in subroutine ERROR. The value of input variable RAT*xxx* must be greater than 1.0. The input variable must be changed. (The nomer *xxx* can refer to any of a number of input variable extensions.)

Message: **INPUT FILE ERROR: NHUB < 2 OR > IXT**

Meaning: An error has been detected in subroutine ERROR. The value of input variable NHUB is less than 2 or greater than PARAMETER IXT. The value of NHUB must be changed, or the code should be recompiled with a larger value of IXT.

Message: **INPUT FILE ERROR: NOUTB < 2 OR > IXT**

Meaning: An error has been detected in subroutine ERROR. The value of input variable NOUTB is less than 2 or greater than PARAMETER IXT. The value of NOUTB must be changed, or the code should be recompiled with a larger value of IJX.

Message: **INPUT FILE ERROR: NBLD < 1**

Meaning: An error has been detected in subroutine ERROR. The value of input variable NBLD is less than 1. There must be at least 1 blade.

Message: **INPUT FILE ERROR: NBLRCS < 2 OR > IJX**

Meaning: An error has been detected in subroutine ERROR. The value of input variable NBLRCS is less than 2 or greater than PARAMETER IJX. The value of NBLRCS must be changed, or the code should be recompiled with a larger value of IJX.

Message: **INPUT FILE ERROR: NPPRC < 2 OR > IJX**

Meaning: An error has been detected in subroutine ERROR. The value of input variable NPPRC is less than 2 or greater than PARAMETER IXT. The value of NPPRC must be changed, or the code should be recompiled with a larger value of IJX.

Message: **INPUT FILE ERROR: RATCLU IS < 0.0 OR > 1.0**

Meaning: An error has been detected in subroutine ERROR. The value of input variable RATCLU is less than 0.0 or greater than 1.0. The value of RATCLU must be changed.

Message: **INPUT FILE ERROR: EPS < 0.0**

Meaning: An error has been detected in subroutine ERROR. The value of input variable EPS is less than 0.0. The value of EPS must be changed.

Message: **COMPLETED MESH ERROR: IL > IXT**

Meaning: An error has been detected in subroutine ERROR. The axial extent of the completed mesh is larger than program array PARAMETER IXT. The grid size must be reduced, or the code should be recompiled with a larger value for IJX.

Message: **COMPLETED MESH ERROR: JL > IXT**

Meaning: An error has been detected in subroutine ERROR. The radial extent of the completed mesh is larger than program array PARAMETER IJX. The grid size must be reduced, or the code should be recompiled with a larger value for IJX.

Message: **COMPLETED MESH WARNING: ADJACENT BLADE ROWS ROTATE IN THE SAME DIRECTION**

Meaning: A possible error has been detected in subroutine ERROR. Based on the geometry of the blades, it has been determined that adjacent blade rows are rotating in the same direction. Standard turbomachinery practice dictates that the blade rows rotate in opposite directions. This may not be a true error, and the program will continue beyond this point.

Message: **COMPLETED MESH WARNING: INTERIOR THETA VALUES EXTEND BEYOND BOUNDARIES**

Meaning: An error has been detected in subroutine ERROR. Circumferential coordinates for points interior to the mesh have been found outside the periodic boundaries of the original single blade passage computational domain. This normally signals a grid generation error, and the grid should be checked very carefully for potential problems.

Message: **COMPLETED MESH WARNING: INTERIOR Z VALUES EXTEND BEYOND BOUNDARIES**

Meaning: An error has been detected in subroutine ERROR. Axial coordinates for points interior to the mesh have been found outside the region between ZINLET and ZEXIT. This normally signals a grid generation error, and the grid should be checked very carefully for potential problems.

Message: **COMPLETED MESH WARNING: INTERIOR R VALUES EXTEND BEYOND BOUNDARIES**

Meaning: An error has been detected in subroutine ERROR. Radial coordinates for points interior to the mesh have been found outside the region between the hub contour and the outer boundary. This normally signals a grid generation error, and the grid should be checked very carefully for potential problems.

Message: **COMPLETED MESH WARNING: NEGATIVE R VALUES**

Meaning: An error has been detected in subroutine ERROR. One or more radial coordinates of the mesh have been found to be negative. This normally signals a geometry input error.

Message: **COMPLETED MESH WARNING: NEGATIVE BLADE**

THICKNESS

Meaning: An error has been detected in subroutine ERROR. The circumferential coordinates of the mesh lead to a negative blade thickness. This can occur when the theta values for the blade surfaces are specified in the reverse order, or if the blade surfaces cross.

Message: **COMPLETED MESH WARNING: RATIO OF ADJACENT BLADE CHORDS > RATBB**

Meaning: An error has been detected in subroutine ERROR. Between the blade rows, the ratio of axial cell spacing, as the grid transitions to/from a blade surface, is greater than the cell spacing ratio specified by RATBB for the region between the blades. This may lead to extra numerical error. This condition can be corrected by adding points to the sparse region(s); removing points from the dense region(s); or by increasing the value of RATBB (which may increase the numerical error).

Other error messages produced by the code or the *SDBLIB* routines are intended to be self explanatory, and are not listed here. Any unexplained errors are almost always due to insufficient array sizes, and the first step in curing the problem should be to increase the array parameters. If this does not work, then it is possible that some small changes in the geometry may be required. There is a known bug in the *CHGRIDV2* code for geometries which have discontinuous definitions (such as a blunt spinner leading edge). The grid generation development slated for later in the program will address this problem.

Table 3.2: *CHGRIV2* Global Array Size Constraints

No Cowl:

1 < NINPLS < (IJX -1 -3*NBLROW)
1 < NEXPLS < (IJX -NINPLS -3*NBLROW)
1 < NBLPTZ < (IJX -NINPLS -NEXPLS)/NBLROW

With a Cowl:

1 < NINPLS < (IJX -1 -1 -3*NBLROW -1 -1)
1 < NEXPLS < (IJX -NINPLS -1 -3*NBLROW -1 -1)
1 < NBLPTZ < (IJX -NINPLS -1 -1 -NEXPLS)/NBLROW
1 < NCOWLI < (IJX -NINPLS -3*NBLROW -1 -NEXPLS)
1 < NCOWLO < (IJX -NINPLS -NCOWLI -3*NBLROW -1 -NEXPLS)

The *PostScript* plots given in the file *fort.15* are intended to give the user a rapid picture of the C-grid mesh quality in terms of the variation

4. *AOA*: 3-D EULER/NAVIER-STOKES SOLVER OPERATING INSTRUCTIONS

4.1 Introduction

This chapter contains the computer program User's Manual for the time-dependent multiple grid block 3-D Euler/Navier-Stokes ducted propfan aerodynamic analysis *ADPAC-AOA*. The flow solver source programs are written in FORTRAN 77, and have been used successfully on Cray UNICOS and IBM VM/CMS mainframe computer systems as well as a Silicon Graphics 4D workstation using a UNIX operating system.

4.2 Compiling the Source Code

An automated compiling system, or *make* facility is provided to simplify the compilation process for UNIX systems. The source code will compile automatically when the command

make

is issued. Although compiling systems vary from machine to machine, the standard make command will usually suffice to create the object code necessary for final compilation. In order to create the executable code, the specific linking command for a given machine must be used. Since the target systems for this code were intended to be a Silicon Graphics workstation, a Cray computer, or an IBM workstation running the AIX operating system, separate facilities are provided for each machine. On a Silicon Graphics (and potentially many other) workstations, simply issue the command:

make

to use the default compiling sequences. On the Cray computer, the command:

make cray

will perform the necessary compilation. On an IBM workstation, the command

make aix

Before attempting to run the program, it is necessary to set the maximum array size required for a given analysis prior to the compilation process. Array dimensions are specified in the program by a PARAMETER statement in each subroutine. The array limits are specified in the file **PARAMETER.INC** by the statements:

PARAMETER(NBMAX = 40)

PARAMETER(NRA3D = 1000000)

PARAMETER(NRA2D = 10000)

PARAMETER(NRA1D = 100)

PARAMETER(NBL2D = 10000)

which will ultimately appear in every subroutine through an **INCLUDE** statement.

The PARAMETER variables are defined as:

NBMAX	Maximum number of grid blocks allowed
NRA3D	Maximum number of all 3-D grid block elements
NRA2D	Maximum number of all 2-D grid block elements
NRA1D	Maximum number of all 1-D grid block elements
NBL2D	Maximum block size of all 2-D grid block elements

The minimum values for each of these array limits for a given grid may be calculated as follows:

$$NRA3D \geq \sum_{i=1}^{i=NBLKS} [(IMX)_i + 1][(JMX)_i + 1][(KMX)_i + 1]$$

$$NRA2D \geq \sum_{i=1}^{i=NBLKS} max[(IMX)_i + 1), (JMX)_i + 1, (KMX)_i + 1]^2$$

$$NRA1D \geq \sum_{i=1}^{i=NBLKS} max[(IMX)_i + 1), (JMX)_i + 1, (KMX)_i + 1]$$

$$NBL2D \geq max\left([(IMX)_i + 1, (JMX)_i + 1, (KMX)_i + 1]\right)^2$$

where $(IMX)_i$, $(JMX)_i$, and $(KMX)_i$ indicate the size of grid block i, and $NBLKS$ is the total number of grid blocks. The limits on NRA1D and NRA2D utilize the largest of all the dimensions of each of the individual grid blocks to eliminate any restrictions on the size of an individual grid block. PARAMETER NBL2D is used for those two-dimensional arrays which are used repeatedly for every grid block, and must therefore be dimensioned large enough to contain the largest 2-D grid block. The requirement that the PARAMETER variables be based on array sizes 1 element larger than the grid dimensions results from the use of phantom points outside the computational domain to impose the numerical boundary conditions. The *AOA* program automatically checks to make sure enough storage is available for all the blocks and issues a fatal error message if the array size is exceeded.

Approximate computational storage and CPU requirements can be estimated for the aerodynamic analysis from the following formulas:

$$\text{CPU sec} \approx 5.0x10^{-5}(\#\ \text{grid points})(\#\ \text{iterations})$$

$$\text{Memory Mw} \approx 3.3x10^{-5}(\#\ \text{grid points})$$

These formulas are valid for a Cray-II computer operating under the UNICOS environment and the cf77 compiler, version 6.0.11 and above. The times reported are for a single processor only, and are not indicative of any parallelization available through the Cray autotasking or microtasking facilities. Steady inviscid flow calculations normally require approximately 2000 iterations to reduce the maximum residual by three orders of magnitude (10^3). Viscous flow calculations generally require 3000 or more iterations to converge. Convergence for a viscous flow case is generally less well behaved than a corresponding inviscid flow calculation, and in many cases, it is not possible to reduce the maximum residual by three orders of magnitude due to oscillations resulting from vortex shedding, shear layers, etc. A determination of convergence for a viscous flow case must often be based on observing the mass flow rate or the power coeffcient and terminating the claculation when these variables no longer change.

The *AOA* program produces output files suitable for plotting using the *PLOT3D* [3], *SURF* [4], and *FAST* [5] graphics software packages developed at the NASA Ames Research Center. *PLOT3D* format data files are written for both absolute and relative flows (see Chapter 2 for a description of the *PLOT3D* format). The user may also elect to have additional *PLOT3D* absolute flow data files output at constant iteration intervals during the course of the solution. These files may be used as instantaneous flow "snapshots" of an unsteady flow prediction.

4.3 Input/Output Files

A sample *AOA* input data file is shown in Fig. 4.1. A brief description of the variables used in the data file follows.

VARIABLE	DESCRIPTION
MACH	Freestream Mach number.
GAMMA	Specific heat ratio (c_p/c_v).
PEXIT	Freestream static pressure. This is applied at the outer exit boundary and integrated radially inward for propellers (**DUCT**=0.0), or applied at the hub and integrated radially outward for compressors (**DUCT**=1.0) to satisfy radial equilibrium.
RPM	Rotational speed (revolutions per minute). This value will be adjusted to match the advance ratio when **ADVR** $\neq$ 0.0.
ADVR	Advance ratio ($J = U/nD$): if = 0.0, rotational speed is determined

```
+--MACH--++-GAMMA--++-PEXIT--++-RPM----++--ADVR--++--DUCT--++--ICOWL-+
  0.7500     1.400   0.68857       0.0  -2.86000        0.0        1.0
+--CFL---++--VIS2--++--VIS4--++---FHLE-++--FHTE--++-BLDROW-++-ALPHA--+
    -5.0     2.000   1.000        21.0     129.0        1.0        0.0
+-FNCMAX-++--REST--++--SAVE--++-FISTEP-++-FNPRNT-++-ROWMAX-++-FINVVI-+
     5.0       1.0       1.0       1.0       1.0        1.0        1.0
+-BLKILJ-++-BLKILK-++-REFINK-++--EPSX--++--EPSY--++--EPSZ--++-FUNINT-+
     1.0       1.0       1.0     -2.00      -2.0      -2.00     9999.0
+-FVISBC-++-FVISRU-++-FVISTI-++-FVISFI-++-FTIMEI-++-FTURBI-++--FTURBB+
     1.0       1.0       1.0       1.0       1.0        1.0     9999.0
+---DIAM-++--TREF--++--PREF--++--RGAS--++---PR---++---PRT--++
     9.0     518.7   2116.80   1716.26       0.7        0.9
```

Figure 4.1: Sample input data file for *AOA* ducted propfan analysis

by RPM.

DUCT Internal flow duct option:

if = 0.0, external flow options are utilized (propeller),

f = 1.0, an internal flow is assumed (compressor). No slip boundary conditions are applied at the outer boundary rather than the characteristic far field condition.

ICOWL Cowl geometry trigger:

if = 0.0, an unducted geometry is assumed,

if = 1.0, duct boundary conditions are applied.

CFL Time step parameter:

if ≤ 0 then this is the CFL number used to determine the calculation time step for local time stepping,

if > 0.0 then this is the CFL number used to determine the calculation time step without local time stepping (time accurate). For steady state calculations, this should not exceed 7.0 for inviscid flow, or 5.0 for viscous flow. For unsteady calculations, values up to 10.0 have been successfully used.

VIS2 Second order damping coefficient (≈ 2.0, divided by 4 in the code).

VIS4 Fourth order damping coefficient (≈ 1.0, divided by 64 in the code).

FHLE Grid index for the hub or spinner leading edge. This variable is only used for viscous solutions to specify at which axial grid index viscous hub boundary conditions should begin. This should be the axial grid index of the true spinner leading edge.

FHTE Grid index for the hub or spinner trailing edge. This variable is only used for viscous solutions to specify at which axial grid index viscous hub boundary conditions should end. This should be the axial grid index of the true hub trailing edge.

BLDROW Blade row parameter. This value determines which blade row is being calculated during a multiple blade row solution (at present this must be equal to 1.0).

ALPHA Angle of attack (degrees). This angle is measured positive in the radial direction for θ=0 (see Fig. 4.2).

FNCMAX Maximum number of time steps to be performed.

REST Restart option parameter:

if = 0.0 no restart file, initialize variables in code,

if = 1.0 a restart file is read.

SAVE Save restart file option parameter:

if = 0.0 no restart file is output at the end of the run,

if = 1.0 a restart file is output at the end of the run.

FISTEP Not used in this version.

FNPRNT Output trigger:

=0.0, no *PLOT3D* output files are written,

=1.0, relative flow and absolute flow *PLOT3D* output files are written.

ROWMAX Maximum number of blade rows in the current solution (must =1.0).

FINVVI Viscous/inviscid solution trigger. This variable determines whether a viscous or inviscid solution is performed:

if = 0, inviscid;

if = 1, viscous.

BLKILJ Endwall boundary layer dissipation elimination parameter. This term controls the number of grid indices in the j direction across which the numerical dissipation is eliminated. This is intended to be used for viscous flows only, and can be any number between 2 and JL, where JL is the maximum index in the j direction. This number should be roughly equal to the number of radial grid lines within the hub or cowl surface boundary layer flow, and must be estimated before runtime.

BLKILK Blade boundary layer dissipation elimination parameter. This term controls the number of grid indices in the k direction across which the numerical dissipation is eliminated. This is intended to be used for viscous flows only, and can be any number between 2 and KL, where KL is the maximum index in the k direction. This number should be roughly equal to the number of radial grid lines within the blade surface boundary layer flow, and must be estimated before runtime.

REFINK Not used in this version.

EPSX Implicit residual smoothing coefficient in the axial direction (≈ 2.0 is a typical value). The absolute value of this term determines the magnitude of the implicit residual smoothing coefficient, while the sign determines the type of smoothing:

if <0.0, then a constant coefficient value equal to the absolute value of

EPSX is used;

if >0.0, then a variable coefficient scheme utilizing **EPSX** is used;

EPSY Implicit residual smoothing coefficient in the radial direction ($\approx$ 2.0 is a typical value). The absolute value of this term determines the magnitude of the implicit residual smoothing coefficient, while the sign determines the type of smoothing:

if <0.0, then a constant coefficient value equal to the absolute value of **EPSY** is used;

if >0.0, then a variable coefficient scheme utilizing **EPSY** is used;

EPSZ Implicit residual smoothing coefficient in the circumferential direction ($\approx$ 2.0 is a typical value). The absolute value of this term determines the magnitude of the implicit residual smoothing coefficient, while the sign determines the type of smoothing:

if <0.0, then a constant coefficient value equal to the absolute value of **EPSZ** is used;

if >0.0, then a variable coefficient scheme utilizing **EPSZ** is used;

FUNINT Number of iterations between *PLOT3D* file interval snapshot output.

FVISBC Trigger for viscous flow solid surface boundary conditions:

=0.0; inviscid flow boundary conditions are used.

=1.0; viscous flow boundary conditions are used. This variable is generally reserved for research calculations only, and it is not recommended that any value other than 1.0 be used.

FVISRU Trigger for viscous flow solver:

=0.0; inviscid flow Runge-Kutta solver is used.

=1.0; viscous flow Runge-Kutta solver is used. This variable is generally reserved for research calculations only, and it is not recommended that any value other than 1.0 be used.

FVISTI Trigger for viscous flow time step evaluation:

=0.0; inviscid flow time step evaluation is used.

=1.0; viscous flow time step evaluation is used. This variable is generally reserved for research calculations only, and it is not recommended that any value other than 1.0 be used.

FVISFI Trigger for viscous flow dissipation and stress evaluation.

=0.0; inviscid flow dissipation operator is used.

=1.0; viscous flow dissipation operator is used. This variable is generally reserved for research calculations only, and it is not recommended that any value other than 1.0 be used.

FTIMEI Number of iterations for time step evaluation update. For best results, this should be 1.0, which implies that the time step is reevaluated at every iteration. However, this value can be increased (< 10) to reduce CPU time by reevaluating the time step every FTIMEI iterations instead.

FTURBI Number of iterations for turbulence model evaluation update. For best results, this should be 1.0, which implies that the turbulence parameters are reevaluated at every iteration. However, this value can be increased (< 10) to reduce CPU time by reevaluating the turbulence quantities every FTURBI iterations instead.

FTURBB Number of iterations before turbulence model is activated: for laminar flow, set to a very large number so the turbulence model is never called. For turbulent flow, the value should be large enough to ensure that the solution has progressed to a reasonable facsimile of the true final flowfield before the turbulence model is activated (> 200).

DIAM True rotor diameter (feet).

TREF Freestream total temperature (degrees Rankine). This value must be relative to a stationary fan. (All calculations are run as if the fan were in a wind tunnel.)

PREF Freestream total pressure (pounds per square foot). This value must be relative to a stationary fan. (All calculations are run as if the fan were in a wind tunnel.)

RGAS Gas constant (foot-pounds per slug degree Rankine).

PR Gas Prandtl number.

PRT Turbulent Prandtl number (0.9 recommended).

4.4 File Names

The Euler/Navier-Stokes solver system produces three primary types of output files: the standard output, plot data files, and restart files. Except for the input file, all other files follow a consistent naming convention based on the UNIX filename structure. The key to this naming process is the *case* name. All filenames have the form:

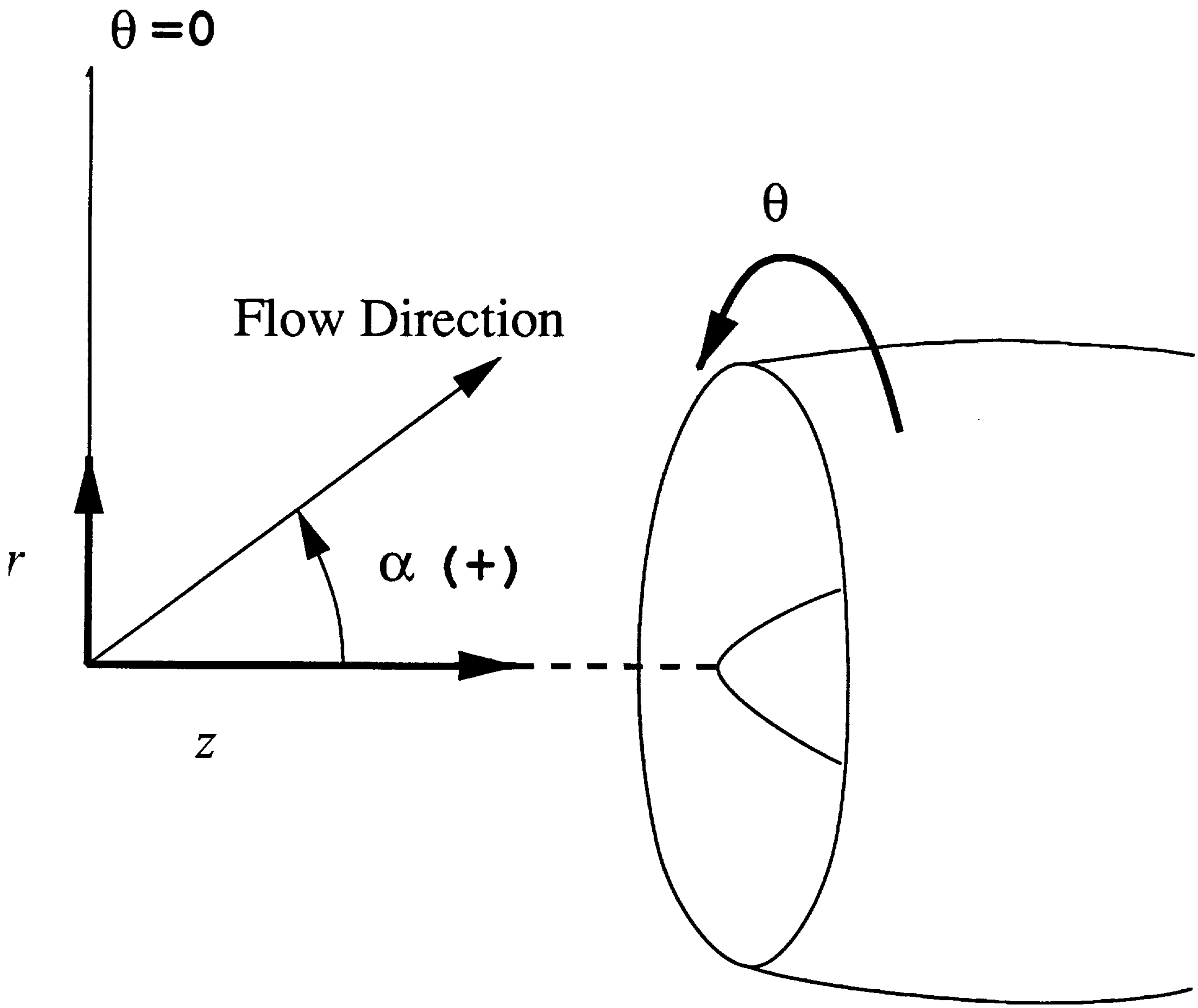

Figure 4.2: Angle of attack geometric definition

Table 4.1: Description of input/output files and UNIX-based filenames for *AOA* Euler/Navier-Stokes solver

Name	Description
case.def	One line file containing the *case* name
case.mesh	Mesh file (*PLOT3D* compatible)
case.p3dabs	Final *PLOT3D* output file (absolute flow)
case.p3drel	Final *PLOT3D* output file (relative flow)
case.p3fr.#	Instantaneous *PLOT3D* interval output file (absolute flow)
	The frame number is given by #.
case.restart.new	New restart file (output by code)
case.restart.old	Old restart file (used as input for restart runs)
case.converge	*FULLPLOT* convergence history plot file (see Appendix)
case.powercoef	*FULLPLOT* power coefficient history plot file (see Appendix)
case.pbpowercoef	*FULLPLOT* per blade power coefficient history plot file (see Appendix)

case.extension

where *case* is a unique name specified by the user to describe the geometry or flow condition being investigated, and *extension* describes the type of file. The *case* name should be available to the code in a file named:

case.def

where **case.def** is a one line file containing the *case* name. A list and description of each of these files is given in Table 4.1 for the *AOA* analysis scheme.

The input and output files are stored in standard ASCII format. All other files utilize the Scientific DataBase Library (SDBLIB) [2] format. The mesh file and *PLOT3D* output files are compatible with the *PLOT3D* multiple grid, binary definition (see Chapter 2).

The standard input and standard output files are directed at runtime as:

aoa < inputfile > outputfile

If a restart run is desired, the user must move the most current output restart file from

case.restart.new

to the default input restart file name

case.restart.old

4.5 Subroutine Description

A list of the 3D Euler/Navier-Stokes solver program subroutines and their functions is given below for reference. A skeleton program flowchart is illustrated in Fig. 4.3.

SUBROUTINE	DESCRIPTION
AOA	Main calling routine.
BCBLA	Blade surface inviscid boundary conditions.
BCBLAV	Blade surface viscous boundary conditions.
BCCOWL	Cowl surface inviscid boundary conditions.

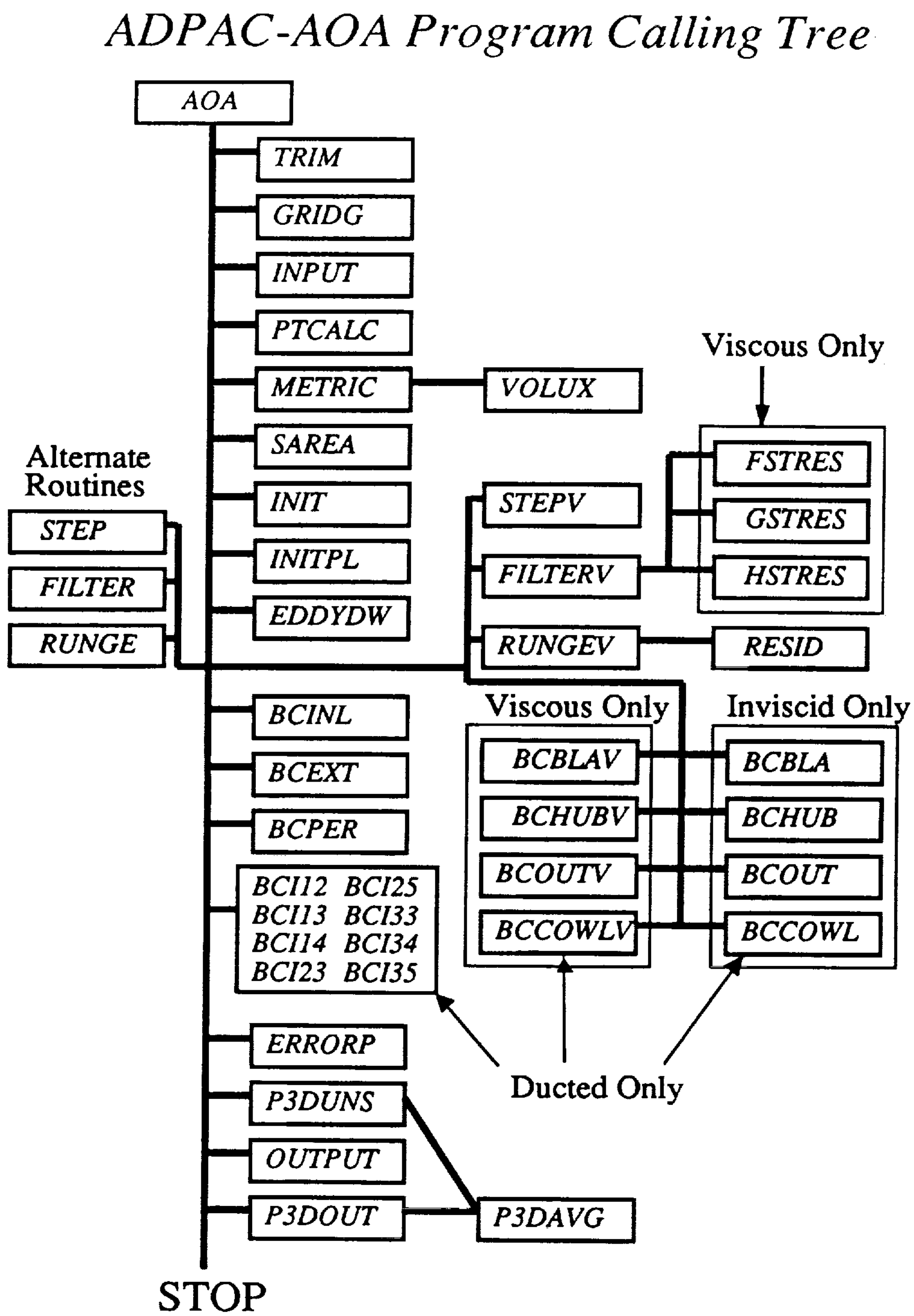

Figure 4.3: Program flowchart for ducted propfan Euler/Navier-Stokes solver

BCCOWLV Cowl surface viscous boundary conditions.

BCEXT Boundary condition routine for exit cells.

BCHUB Hub surface inviscid boundary condition.

BCHUBV Hub surface viscous boundary condition.

BCI12 Boundary condition routine coupling block 1 and 2.
Grid block numbers are illustrated on Fig. 2.1.

BCI13 Boundary condition routine coupling blocks 1 and 3.
Grid block numbers are illustrated on Fig. 2.1.

BCI14 Boundary condition routine coupling blocks 1 and 4.
Grid block numbers are illustrated on Fig. 2.1.

BCI23 Boundary condition routine coupling blocks 2 and 3.
Grid block numbers are illustrated on Fig. 2.1.

BCI25 Boundary condition routine coupling blocks 2 and 5.
Grid block numbers are illustrated on Fig. 2.1.

BCI33 Boundary condition routine along C-grid branch cut.

BCI34 Boundary condition routine coupling blocks 3 and 4.
Grid block numbers are illustrated on Fig. 2.1.

BCI35 Boundary condition routine coupling blocks 3 and 5.
Grid block numbers are illustrated on Fig. 2.1.

BCI45 Boundary condition routine coupling blocks 4 and 5.
Grid block numers are illustrated on Fig. 2.1.

BCINL Boundary condition routine for inlet cells.

BCOUT Boundary condition routine for free-flow outer boundary.

BCOUTV	Boundary condition routine for viscous outer boundary.
BCPER	Boundary condition routine for periodic cells.
CONATAN	Continuous tangent function evaluation routine.
CONVAS	Array storage conversion routine used in conjunction with SDBLIB machine-independent input/output routines.
CURPLT	Curve plotting routine.
DEBOUT	Debug output routine.
EDDYDW	Turbulence model evaluation routine.
ERRORP	Convergence-checking routine for Runge-Kutta solver for propfan calculations.
FILNAM	Input/output filename contruction routine.
FILTER	Artificial dissipation routine.
FILTERV	Artificial dissipation routine for viscous flows.
FIXUP	Routine to correct corner phantom celss so output averages and viscous stress evaluations are correct.
FPINIT	Plot initialization routine.
FSTRES	Viscous stress evaluation along i coordinate direction.
GRAPLT	Gray-shaded plot routine.
GRIDG	Routine to read and set up the grid.
GSTRES	Viscous stress evaluation along j coordinate direction.
HSTRES	Viscous stress evaluation along k coordinate direction.
INIT	Routine to initialize flowfield.
INITPL	Plot initialization routine.

INPUT Routine to read in input data and set up reference values.

INTCHA Integer to character conversion routine.

LOGOPL Logo plotting routine.

LTRIM Routine to determine the length of a trimmed character string.

METRIC Routine to calculate cell volumes and surface normals.

OUTPUT Routine to print output and save restart files.

P3DAVG Routine to average data for a *PLOT3D* output file.

P3DOUT Routine to output *PLOT3D* final data file.

P3UNS Routine to output *PLOT3D* interval data file.

PTCALC Routine to determine inlet total pressure profile.

RELCHA Real to character conversion routine.

RESID Implicit residual smoothing routine.

RUNGE Runge-Kutta inviscid flux calculation routine.

RUNGEV Runge-Kutta viscous flux calculation routine.

SAREA Routine to determine metric area terms.

STEP Routine to determine time step for inviscid flow.

STEPV Routine to determine time step for viscous flow.

TRIM Routine to delete trailing blanks from a character string.

VOLUX Routine to calculate an individual cell volume.

It should also be mentioned that a number of routines from the SDBLIB library are also included in the source code distribution, but are not defined in detail here.

4.6 Error Messages

AOA has an extended internal error checking facility which is intended to warn the user of potential problems during the course of a calculation. This section describes the meaning of the error and warning messages produced by *AOA* and possible courses of action to correct the errors.

Message: **ERROR DETECTED IN AOA.f UNABLE TO OPEN FILE**

Meaning: An error has occurred in subroutine AOA. While attempting to open a file for reading, an error has occurred. Make sure that the indicated file is available to the code for input.

Message: **ERROR DETECTED IN AOA.f ABORT! - NUMBER OF GRID BLOCKS IN MESH FILE EXCEEDS PROGRAM ARRAY PARAMETER NBMAX**

Meaning: An error has occurred in subroutine AOA. While attempting to read in the grid file, the program has determined that the number of grid blocks will exceed program array dimension NBMAX. The user must recompile the code with a larger value of PARAMETER NBMAX. Following the error message, the minimum permissable value of NBMAX for the current grid is listed.

Message: **ERROR DETECTED IN AOA.f ABORT! - 3-D POINTER SIZE EXCEEDS PROGRAM ARRAY PARAMETER NRA3D**

Meaning: An error has occurred in subroutine AOA. While attempting to read in the grid file, the program has determined that the combined total 3-D array elements required for all the grid blocks will exceed program array PARAMETER NRA3D. The user must recompile the code with a larger value of PARAMETER NRA3D. Following the error message, the minimum permissable value of NRA3D for the current grid is listed.

Message: **ERROR DETECTED IN AOA.f ABORT! - 2-D POINTER SIZE EXCEEDS PROGRAM ARRAY PARAMETER NRA2D**

Meaning: An error has occurred in subroutine AOA. While attempting to read in the grid file, the program has determined that the combined total 2-D array elements required for all the grid blocks will exceed program array PARAMETER NRA2D. The user must recompile the code with a larger value of PARAMETER NRA2D. Following the error message, the minimum permissable value of NRA2D for the current grid is listed.

Message: **ERROR DETECTED IN AOA.f ABORT! - 1-D POINTER SIZE EXCEEDS PROGRAM ARRAY PARAMETER NRA1D**

Meaning: An error has occurred in subroutine AOA. While attempting to read in the grid file, the program has determined that the combined total 1-D array elements required for all the grid blocks will exceed program array PARAMETER NRA1D. The user must recompile the code with a larger value of PARAMETER NRA1D. Following the error message, the minimum permissable value of NRA1D for the current grid is listed.

Message: **ERROR DETECTED IN AOA.f NUMBER OF BLOCKS IN RESTART FILE DOES NOT MATCH THE NUMBER OF GRID BLOCKS**

Meaning: An error has occurred in subroutine AOA. While attempting to read in the restart file, the program has determined that the number of grid blocks specified in the grid and restart files are inconsistent. It is likely that the restart file was not generated from the given grid file. The user should check the origin of the grid and restart files to make sure that they are compatible.

Message: **ERROR DETECTED IN AOA.f RESTART FILE BLOCK SIZE DOES NOT MATCH CORRESPONDING GRID BLOCK SIZE**

Meaning: An error has occurred in subroutine AOA. While attempting to read in the restart file, the program has determined that the grid block sizes indicated in the restart file are not consistent with the grid block sizes indicated in the mesh file. The user should check the origin of the grid and restart files to make sure that they are compatible.

Message: **ERROR - NBLKS DOES NOT EQUAL NPASG * NBPPAS**

Meaning: An error has occurred in subroutine AOA. While checking the number of grid blocks specified in the mesh file, the code has determined that the total number of grid blocks is not consistent with the requirements for the allowable ducted or unducted, single passage or full rotor grid configurations. The most common cause for this error is the input specification of a ducted flow for an unducted grid or specifying and angle of attack for a single passage grid. The user should check the input parameters COWL and ALPHA.

Message: **P3DUNS: UNABLE TO OPEN FILE**

Meaning: An error has occurred in subroutine P3DUNS. While attempting to open a file for *PLOT3D* interval (unsteady) output, an error has occurred. It is possible that the file system is full, or that the user does not have write permission for this file, or that the user does not have write permission for this directory.

Message: **P3DOUT: UNABLE TO OPEN FILE**

Meaning: An error has occurred in subroutine P3DOUT. While attempting to open a file for *PLOT3D* final output, an error has occurred. It is possible that the file system is full, or that the user does not have write permission for this file, or that the user does not have write permission for this directory.

Message: **ERROR DETECTED IN AOA.f UNABLE TO OPEN FILE**

Meaning: An error has occurred in subroutine AOA. While attempting to open a file for reading, an error has occurred. Make sure that the indicated file is available to the code for input.

Message: **CONVAS: ERROR - CANNOT DETERMINE CONVERSION PROCESS**

Meaning: An error has occurred in subroutine CONVAS. During the course of a conversion from one array structure to another, CONVAS has discovered that the input and output array sizes are inconsistently specified. This error should never occur.

Message: **WARNING! FHLE < 1 ; HUB LEADING EDGE INDEX SET TO #**

Meaning: Subroutine INPUT has detected an error in the input file. The value of FHLE is less than one, and has been reset to a value of 1.0.

Message: **WARNING! FHLE > FHTE OR FHLE=FHTE**

Meaning: Subroutine INPUT has detected an error in the input file. The values of FHLE and FHTE are inconsistent. FHLE is reset to 1.0 and FHTE is reset to the last axial grid index.

Message: **WARNING! FHTE < 2 ; HUB TRAILING EDGE INDEX SET TO #**

Meaning: Subroutine INPUT has detected an error in the input file. The value of FHTE is less than two, and has been reset to the last axial grid index.

Message: **WARNING! HUB LEADING EDGE INDEX FHLE > IL(1)**

Meaning: Subroutine INPUT has detected an error in the input file. The value of FHLE is greater than the axial grid size, and has been reset to a value of 1.0.

Message: **WARNING! HUB TRAILING EDGE INDEX FHTE > IL(1)**

Meaning: Subroutine INPUT has detected an error in the input file. The value of FHTE is greater than the axial grid size, ans has been reset to the last axial grid index.

Message: **WARNING! *TERM* DOES NOT MATCH GRID FILE**

Meaning: Subroutine INPUT has detected an error in the restart file. The value of *TERM* (where *TERM* is any of a number of variables) in the restart file does not match the corresponding value in the grid file. This may or may not be a significant error. This error can occur if the grid and restart files are not compatible. The value in the grid file takes priority.

Message: **WARNING! *TERM* DOES NOT MATCH FLOW FILE**

Meaning: Subroutine INPUT has detected an error in the input file. The value of *TERM* (where *TERM* is any of a number of variables) in the input file does not match the corresponding value in the restart file. This may or may not be a significant error. The value in the input file takes priority.

Other error messages produced by the code or the *SDBLIB* routines are intended to be self explanatory, and are not listed here. Unexpected errors such as non-convergent results and or unexpected math library errors are often traced to problems associated with grid resolution, grid skewness, insufficient damping, or excessive time increments. The standard values listed in the input variable descriptions should suffice to provide guidance for such cases.

5. *ADPAC* TOOL PROGRAMS OPERATING INSTRUCTIONS

5.1 Introduction

Included with the *ADPAC* distribution are a number of tools which are designed to aid the user in developing solutions for unducted and ducted propfans at angle of attack. The specific nature of each tool is listed in the sections below. In general, the tools programs have array size limits based on a PARAMETER statement in the program such as:

PARAMETER(IMX=100, JMX=51, KMX=31, NBLKS=50)

The parameters IMX, JMX, and KMX determine the maximum size of the largest grid block in the axial, radial, and circumferential directions, respectively, for a single-passage grid or flow solution. The parameter NBLKS determines the maximum number of blocks for the full rotor grid or flow file. A fatal error message is issued when these limits are exceeded.

The file naming procedures described in the previous two chapters are also extended to the tools programs. All file names are in the form

case.extension

where *case* is the case name defined in the file

case.def

and *extension* is a unique file extension which indicates the contents of the file.

The operation of each tool is generally self explanatory, therefore, only a brief description of the function of each code is given below.

5.2 *ROTCGRID*: Full Rotor Grid Rotation Routine

The *ADPAC* program *ROTCGRID* was developed in order to duplicate a single blade passage grid into a grid suitable for a full rotor solution. *ROTCGRID* searches for a single blade passage grid based on the file naming procedure discussed in the previous chapters, and performs a duplication and rotation process on this initial grid in order to generate a full rotor multiple-block grid. Of course, the grid generation program *CHGRIDV2* must have been previously run to generate the single passage mesh file. This assumes that the geometry is spatially periodic, even if the resulting flowfield is not. Thus, this scheme is only useful for axisymmetric cowl and hub geometries. This limitation is also present in the aerodynamic analysis provided by *AOA*. Presumably, if some alternate grid generation scheme is developed to generate a similar multiple-block grid for a nonaxisymmetric cowl geometry, then the flow solver *AOA* could be modified to provide solutions for nonaxisymmetric cowl geometries.

A summary of the UNIX system filenames and a description of their contents is given in Table 5.1. Of course, the output mesh file *case*.meshrot must be renamed *case*.mesh before it may be used as input for the aerodynamic analysis.

Table 5.1: Description of input/output files and UNIX-based filenames for *ROTCGRID* grid rotation program

Name	Description
case.mesh	Original grid from *CHGRIDV2*
case.meshrot	Full rotor mesh file

Table 5.2: Description of input/output files and UNIX-based filenames for *ROTCFLOW* flow rotation program

Name	Description
case.restart.old	Original flow file from *AOA*
case.restart.oldrot	Full rotor flow file

5.3 *ROTCFLOW*: Full Rotor Flow Rotation Routine

The *ADPAC* program *ROTCFLOW* was developed in order to duplicate a single blade passage flow file into a flow file suitable for a full rotor restart solution. *ROTCFLOW* searches for a single blade passage mesh and flow restart file based on the file naming procedure discussed in the previous chapters, and performs a duplication and rotation process on this initial flow in order to generate a full rotor multiple-block flow restart file. This code is therefore useful for constructing full rotor initial data files for unsteady predictions initiated from a single-passage steady state solution. Of course this presumes that a restart file containing the single passage flow data from *AOA* has been previously generated.

A summary of the UNIX system filenames and a description of their contents is given in Table 5.2. Of course, the output flow file *case*.flowrot must be renamed *case*.restart.old before it may be used as input for the aerodynamic analysis.

Table 5.3: Description of input/output files and UNIX-based filenames for *AOAPLOT* plotting program

Name	Description
case.mesh	Original mesh file from *CHGRIDV2* or *ROTCGRID*
case.restart.old	Flow restart file from *AOA*
case.p3dfr.n	*PLOT3D* file containing instantaneous unsteady output
case.aoaplot	*AOAPLOT* PostScript output file

5.4 *AOAPLOT*: Automated *PostScript* Plotting Routine

The *ADPAC* program *AOAPLOT* was developed to provide an automated plotting facility to analyze airfoil and cowl performance data for both steady and unsteady numerical simulations. The *AOAPLOT* program is interactively menu driven, and prompts the user for the type of plot desired. A series of plots are available for the input geometry, grids, steady state solutions, and unsteady simulations for which time-interval output are available. All plot output is based on the *PostScript* page description language, which is suitable for a large number of printing and on-screen viewing facilities.

A summary of the UNIX system filenames and a description of their contents is given in Table 5.3.

5.5 *FULLPLOT*: *PostScript* X-Y Plotting Routine

The *ADPAC* program *FULLPLOT* was developed to provide an automated plotting facility for simple x-y type plots. The *ADPAC* program *AOA* produces three files which monitor convergence history, power coefficient history, and per blade power coefficient history. This data is output in the files labelled *case*.converge, *case*.powercoef, and *case*.pbpowercoef, respectively. These files are written in a format which *FULLPLOT* will use to create *PostScript* based plots. The execution of this program is extremely simple and can be invoked by the command:

fullplot <*filename*

where *filename* is one of the three files mentioned above.

Array limits in the *FULLPLOT* program are determined by the PARAMETER statement:

PARAMETER(MAXCUR=50, MAXPTS=5000)

The parameter MAXCUR determines the maximum number of cirves that *FULLPLOT* can plot, while MAXPTS determines the maximum number of points per curve.

Input to *FULLPLOT* is directed upon execution, and this program can therefore easily be used for other two-dimensional plotting purposes. *FULLPLOT* produces a *PostScript* plot file labelled *fort.15*, which can be sent directly to a *PostScript* printer, or previewed on a compatible device. The *FULLPLOT* input file may be modified by the user to construct the final plots in a number of forms. A sample *FULLPLOT* input file is given in Fig. 5.1. The *PostScript* plot for this input file is given in Fig. 5.2. Each of the various line and symbol types, as well as the *FULLPLOT* grayscale shading are illustrated in the sample plot.

The actual variables in the input file are free format. A description of each of the input PARAMETERS is given in the paragraphs below.

```
TITLE---- 3 LINES MAXIMUM -----------------------------------------------
'This is the first title line'
'This is the second title line'
'This is the third title line'
LVH --- VERTICAL (0) OR HORIZONTAL (1) PLOT ----------------------------------
0
LEGTYP----DXLEGI----DYLEGI----DXBOXI--(DXLEGI, DYLEGI INCHES FROM ORIGIN)----
2         0.25      7.0       3.5
LOGO------DXLOGOI---DYLOGOI-----------(LOGO =0 NO LOGO, DX, DY IN INCHES)----
1         -1.3      -1.5
XLABEL-----------------------------------------------------------------------
'X axis Label'
NINCX-----XSMIN-----XSMAX-----LOCXAX----NXSIG--------------------------------
5          0.0       1.0       0         2
YLABEL-----------------------------------------------------------------------
'Y Axis Label'
NINCY-----YSMIN-----YSMAX-----LOCYAX----NYSIG--------------------------------
5         -1.0       3.0       0         2
NUMCUR-----------------------------------------------------------------------
14
CURVE DATA ----#PTS,LTYPE,STYPE,LEGEND,GRAYPLT,GRAYSCALE,LEGLABEL------------
            2 1 1 1 0 0.0 'Line 1 Symbol 1'
  0.1      0.1
  0.1      0.9
            2 2 2 1 0 0.0 'Line 2 Symbol 2'
  0.2      0.1
  0.2      0.9
            2 3 3 1 0 0.0 'Line 3 Symbol 3'
  0.3      0.1
  0.3      0.9
            2 4 4 1 0 0.0 'Line 4 Symbol 4'
  0.4      0.1
  0.4      0.9
            2 5 5 1 0 0.0 'Line 5 Symbol 5'
  0.5      0.1
  0.5      0.9
            2 6 6 1 0 0.0 'Line 6 Symbol 6'
  0.6      0.1
  0.6      0.9
            2 7 7 1 0 0.0 'Line 7 Symbol 7'
  0.7      0.1
  0.7      0.9
            2 8 8 1 0 0.0 'Line 8 Symbol 8'
  0.1     -0.1
  0.1     -0.9
            2 9 9 1 0 0.0 'Line 9 Symbol 9'
  0.2     -0.1
  0.2     -0.9
            2 10 10 1 0 0.0 'Line 10 Symbol 10'
  0.3     -0.1
  0.3     -0.9
            2 11 11 1 0 0.0 'Line 11 Symbol 11'
  0.4     -0.1
  0.4     -0.9
            2 12 12 1 0 0.0 'Line 12 Symbol 12'
  0.5     -0.1
  0.5     -0.9
            2 13 13 1 0 0.0 'Line 13 Symbol 13'
  0.6     -0.1
  0.6     -0.9
            5 1 0 1 1 0.5 'Gray Shaded - Grayscale=0.5'
  0.8     -0.1
  0.9     -0.5
  0.8     -0.9
  0.7     -0.5
  0.8     -0.1
```

Figure 5.1: Sample input data file for *FULLPLOT* plotting program

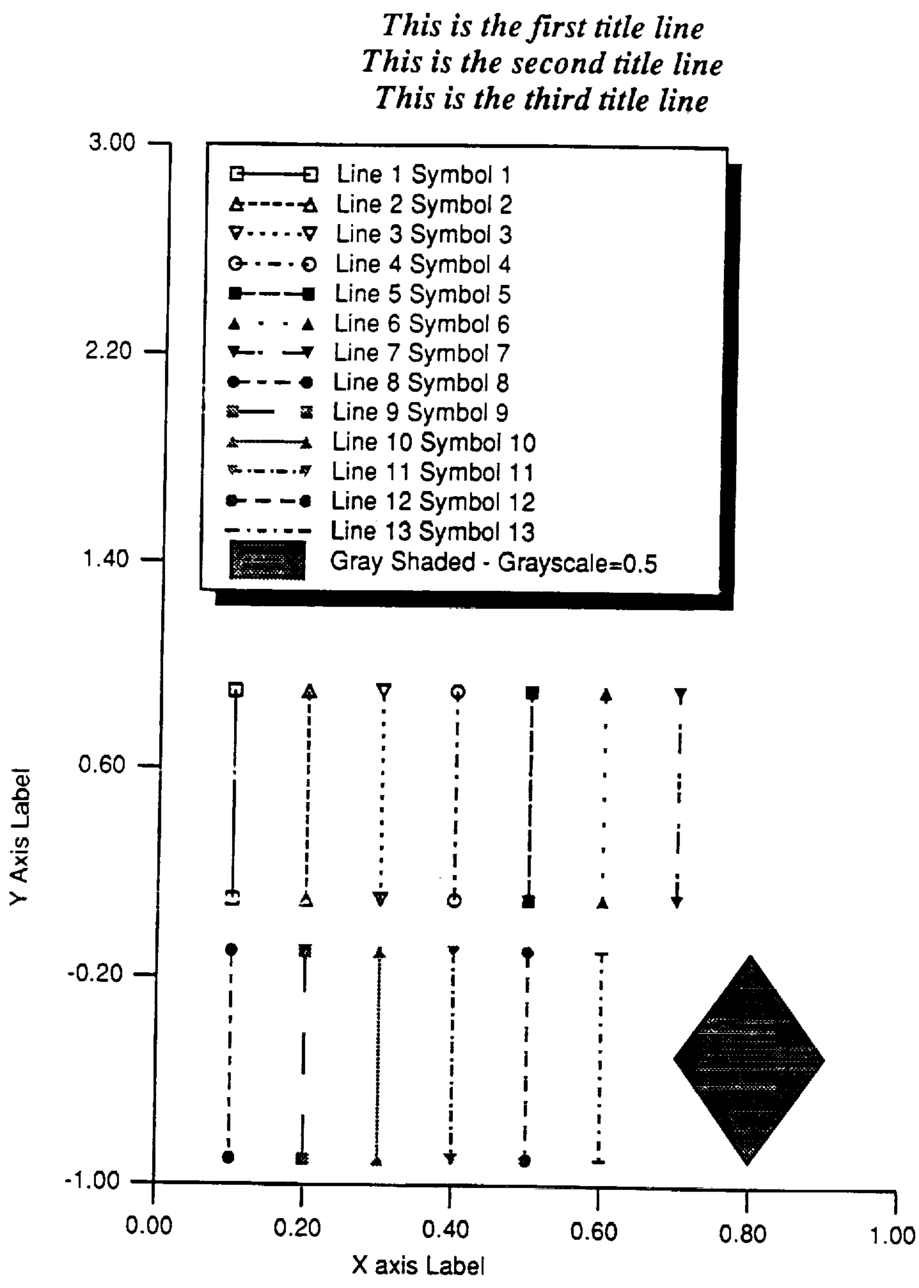

Figure 5.2: Sample PostScript x-y plot from *FULLPLOT* plotting program

VARIABLE	DESCRIPTION
TITLE	Three lines of 80 character title data follow the TITLE header line. The 3 strings will be centered above the plot, and are plotted in the same order as they are input. Each title string must be in quotes.
LVH	Vertical/horizontal plot trigger: if = 1, the plot y axis is vertical on a standard $8\frac{1}{2}$ x 11 page (portrait mode), if = 2, the plot y axis is horizontal on a standard $8\frac{1}{2}$ x 11 page (landscape mode).
LEGTYP	Legend type plot trigger: if = 0, no legend is plotted, if = 1, a plain legend is plotted, if = 2, a shadow box legend is plotted,
DXLEGI	Legend placement variable. This variable determines the actual physical length, in inches, from the lower left hand legend corner to the x axis.
DYLEGI	Legend placement variable. This variable determines the actual physical length, in inches, from the lower left hand legend corner to the y axis.
DXBOXI	Legend box length parameter. The legend height is predetermined, but the legend box length is not. This variable permits user control of this fuction. This variable controls the actual physical length of the

legend box in inches.

LOGO Logo plotting trigger:

if = 0, no logo is plotted,

if = 1, the NASA logo is plotted.

DXLOGOI Logo placement variable. This variable determines the actual physical length, in inches, from the lower left hand logo corner to the x axis.

DYLOGOI Logo placement variable. This variable determines the actual physical length, in inches, from the lower left hand logo corner to the y axis.

XLABEL X axis label. This is an 80 character string which will be centered below the x axis. The string must be in quotes.

NINCX Number of scale increments along the x axis. The x scale is determined by the number of increments and the minimum and maximum x axis scale values XSMIN, and XSMAX. The scale increment is then simply (XSMAX-XSMIN) / (NINCX-1)

XSMIN X axis minimum scale value (see NINCX).

XSMAX X axis maximum scale value (see NINCX).

LOCXAX Not used.

NXSIG Number of significant digits past the decimal point in x axis scale markers.

YLABEL Y axis label. This is an 80 character string which will be centered along the y axis. The string must be in quotes.

NINCY Number of scale increments along the y axis. The y scale is determined by the number of increments and the minimum and maximum y axis scale values YSMIN, and YSMAX. The scale increment is then simply (YSMAX-YSMIN) / (NINCY-1)

YSMIN X axis minimum scale value (see NINCX).

YSMAX X axis maximum scale value (see NINCX).

LOCYAX Not used.

NYSIG Number of significant digits past the decimal point in y axis scale markers.

NUMCUR Number of separate sets of data (curves) to be drawn. Each curve is represented by a set of x-y data points which can be represented by points, lines, or as a shaded region on the plot.

The remainder of the *FULLPLOT* data set is the actual data to be plotted. The following data information must be repeated for each separate curve to be plotted. Each set of data (curve) begins with a header line with the following information:

#PTS Number of x-y data pairs in the curve.

LTYPE Line type used to draw the curve. A curve may be represented by lines, symbols, shaded regions, or all of the above.

The line types are defined as:

LTYPE=0 is no line (symbol only),

LTYPE=1 is a solid line,

LTYPE=2 is a medium dashed line,

LTYPE=3 is a short dashed line,

LTYPE=4 is a medium chained dash line,

LTYPE=5 is a very long dashed line.

LTYPE=6 is a very short dashed line.

LTYPE=7 is a staggered chain dashed line.

LTYPE=8 is a alternate dash length line.

LTYPE=9 is a very wide spaced dashed line.

LTYPE=10 is a very short spaced short dashed line.

LTYPE=11 is a tight staggered chained dashed line.

LTYPE=12 is a very tight staggered chained dashed line.

LTYPE=13 is a very short dash chained line.

STYPE Symbol type used to draw the curve. A curve may be represented by lines, symbols, shaded regions, or all of the above.

The symbol types are defined as:

STYPE=0 is no symbol (line only),

STYPE=1 is an open square,

STYPE=2 is an open triangle pointing up,

STYPE=3 is an open triangle pointing down,

STYPE=4 is an open circle,

STYPE=5 is an black filled square,

STYPE=6 is an black filled triangle pointing up,

STYPE=7 is an black filled triangle pointing down,

STYPE=8 is an black filled circle,

STYPE=9 is an gray filled square,

STYPE=10 is an gray filled triangle pointing up,

STYPE=11 is an gray filled triangle pointing down, STYPE=12 is an gray filled circle.

STYPE=13 is a horizontal bar.

LEGEND A trigger to indicate whether the current curve should be included in the legend: if = 0, don't include this curve in the legend, if = 1, include this curve in the legend.

GRAYPLT A trigger to indicate whether the current curve should be plotted as a shaded region. If this is desired, the data will be interpreted as a closed curve which will have a shaded interior based on the shading value GRASCALE:

if = 0, don't treat this curve as a shaded region,

if = 1, treat this curve as a shaded region.

GRASCALE Shaded region shading value. This value must lie between 0.0 and 1.0. The value 0.0 represents true black, while a value of 1.0 represents true white. All other values in between represent increasing lighter shades of gray. This value is only used when GRAYPLT=1.

LEGLABEL A character string (must be in quotes) indicating the legend label for this curve.

The remaining lines represent the #PTS pairs of x,y data in free format. One data set is given per line for the given curve.

REFERENCES

[1] Hall, E. J., Delaney, R. A., and Bettner, J. L., "Investigation of Advanced Counterrotation Blade Configuration Concepts for High Speed Turboprop Systems: Task II - Unsteady Ducted Propfan Analysis, Final Report", NASA CR 187106, NASA Contract NAS3-25270, 1991.

[2] Whipple, D., "BDX-Binary Data Exchange Preliminary Information", NASA-Lewis Research Center, 1989.

[3] Walatka, P. P., and Buning, P. G., "PLOT3D User's Manual,", rough draft for NASA TM, 1988.

[4] Plessel, Todd, "SURF User's Guide,", NASA Ames Research Center, 1988.

[5] Walatka, P. P., and Buning, P. G., "FAST", NASA Ames Research Center, 1990.

[6] Hall, E. J., Delaney, R. A., and Bettner, J. L., "Investigation of Advanced Counterrotation Blade Configuration Concepts for High Speed Turboprop Systems: Task I - Ducted Propfan Analysis", NASA CR 185217, NASA Contract NAS3-25270, 1990.

APPENDIX A. *ADPAC* DISTRIBUTION AND DEMONSTRATION INSTRUCTIONS

A.1 Introduction

This appendix describes the commands necessary to extract the *ADPAC* source code distribution from the standard distribution and run a complete test case for a ducted fan at angle of attack. The standard *ADPAC* distribution is normally a compressed *tar* file which can be decodeded into the various parts by a sequence of commands on any standard UNIX system. The sequence listed below is intended to guide the user through the setup from the standard distribution up to and including a complete demonstration of a time-dependent aerodynamic solution for a ducted propfan at angle of attack.

A.2 Extracting the Source Files

The *ADPAC* programs are distributed as a compressed *tar* file named

adpac.tar.Z

This tar file requires roughly 2.5 megabytes of disk space. It should be possible to extract and run the code on any standard UNIX system from this distribution file.

The first step necessary to extract the *ADPAC* programs is to uncompress the *tar* file with the command:

uncompress adpac.tar.Z

This operation essentially replaces the compressed file *adpac.tar.Z* with an uncompressed file *adpac.tar*. The uncompressed *tar* file requires approximately 7.5 megabytes of disk space.

The next step is to extract the individual files and directories from the *adpac.tar* file. The **tar** command will create a subdirectory named *adpac* in the current directory, so the user should move the *adpac.tar* file to a suitable initial directory. Once the *tar* file is properly placed, the *ADPAC* distribution may be extracted with the command

tar xvof adpac.tar

(On some systems **tar xvf adpac.tar** may be sufficient.) Execution of the command ls -l will verify that the *adpac* directory has been created. The complete extraction process will require about 10 megabytes of disk space.

A.3 Compiling the Source Code

After extracting the source files, the user is naturally interested in compiling the source files for execution. A UNIX-compatible *Make* facility is provided for each of the *ADPAC* programs. The *Makefile* which governs the compilation process is necessarily machine-dependent and requires that the user select from one of a number of preconfigured systems. The systems which are imediately available are:

iris Silicon Graphics Iris workstation

cray Cray Computer Inc. supercomputer

aix IBM Aix operating system UNIX workstation

If no *system* is specified, then the *iris* system is assumed. The code will also compile on other system with minor *Makefile* modifications. The machine dependence of the compilation process is inherently tied to the use of the Scientific DataBase Library routines for binary file input/output.

In order to begin the compilation, it is first necessary to enter the *adpac* directory with the command:

cd adpac

At this point, several files and directories will be available. By entering the command ls, a listing of the individual directories can be obtained. The output of the ls command will look something like:

```
demo/    Makefile    manual/    report/    src/    tools/
```

A description of each of these listings is given below:

demo This directory contains several geometry and flow input files for generating sample runs of the *ADPAC* codes.

Makefile This file is the global *Makefile* for the compiling system.

manual This directory contains the *LaTeX* source code for this manual. If *LaTeX* is installed on your system, it is possible to reproduce this document (excluding figures) with the command **latex manual**. The resulting device independent file *manual.dvi* may then be converted to *PostScript* or previewed on screen through a number of widely available routines.

report This directory contains the *LaTeX* source code for the final report outlining the technical details of the *ADPAC* codes. If *LaTeX* is installed on your system, it is possible to reproduce the final report (excluding figures) with the command **latex finalrep**. The resulting device independent file *finalrep.dvi* may then be converted to *PostScript* or previewed on screen through a number of widely available routines.

src This directory contains all the FORTRAN source code for the *ADPAC* programs *CHGRIDV2* and *AOA*.

sdblib This directory contains the various machine-dependent files for the Scientific DataBase Library routines.

tools This directory contains the source code for the *ADPAC* tools programs.

It is now possible to compile the *ADPAC* codes and tools. By issuing the command

make *system*

where *system* indicates the current computing platform (iris, cray, or aix) described above. From the main directory, issuing the **make** command compiles all source and tools programs. Individual *Makefiles* are also included in each source and tools subdirectory to permit separate compilation of the individual codes. Special options may apply to the individual source programs and users are encouraged to read the manual sections for the individual codes to tailor the compilation process for their own needs. The compilation of the executable modules will require roughly 17 megabytes of disk space.

A.4 Running the Distribution Demonstration Test Case

Once the *make* facility has properly completed compiling the *ADPAC* source code distribution, it is possible to run the test cases provided with the standard distribution. It is recommended that the sample cases be tested to verify proper compilation and extraction of the *ADPAC* distribution.

In order to run the demonstration cases, it is necessary to begin in the *demo* directory. At this point, the *demo* directory may be entered by issuing the command

cd demo

Both ducted and unducted test cases are provided to illustrate the operation of the codes and the sequence of events leading up to a full angle of attack solution. The commands needed to run either demo are similar, so only the ducted test case will be outlined here.

In the demo directory, an ls command will indicate that the following subdirectories are avaiable:

```
ducted/    unducted/
```

These subdirectories obviously contain the ducted and unducted demonstration cases, respectively. To run the ducted case, enter the *ducted* subdirectory by issuing the command **cd ducted**. Now, the ls command reveals:

```
case.def   ducted.ggenin   ducted.steady.input   ducted.unsteady.input
```

The file *case.def* is a single line file containing the string *ducted* which will define the *case* name used to name all subsequent files. By default then, the initial grid generation input file is named *ducted.ggenin*, and the grid generation program *CHGRIDV2* will automatically search for this file upon execution. The grid generation process is launched by issuing the command

../../src/chgridv2/chgridv2

Upon execution, several messages will appear on the screen inidicating the progress of the mesh construction. The terminating message is **PROGRAM COMPLETED NORMALLY**. Now, an ls command reveals the existence of two new files:

```
ducted.ggenout    ducted.mesh
```

The desired multiple-block mesh is now contained in *ducted.mesh*, and may be viewed using *PLOT3D*.

The next step is to generate a steady flow solution for our newly-created mesh. The steady flow solution is controlled by the aerodynamic analysis program *AOA* and the input file *ducted.steady.input*. The steady state solution is generated by issuing the command

```
../../src/aoa/aoa <ducted.steady.input >ducted.steady.output
```

It should be mentioned that the standard input and output are redirected in this case, which implies that the *AOA* input and output file names are not required to follow the *case* naming convention described in this report. The computation time required to generate the steady state solution may take up to an hour on a workstation-class computer. Once the steady flow solution has been generated, the ls command will reveal several new files:

```
ducted.restart.new     ducted.p3drel     ducted.p3dabs
```

The file *ducted.restart.new* contains the restart file necessary to continue this run from the point of termination. The file *ducted.p3dabs* and *ducted.p3drel* contain the absolute ansd relative flow *PLOT3D* flow variable information, respectively. It may be of interest to examine the steady flow results with *PLOT3D* at this point (see Ref. [3] for details).

Our next step is to construct a full rotor mesh and flowfield from the results of the single blade row steady flow calculation. The first step is to move the steady flow restart file so the codes will recognize this data as the most recent run. This may be accomplished by issuing the command

mv ducted.restart.new ducted.restart.old

Now it is possible to utilize the *ADPAC* tools programs *ROTCGRID* and *ROTCFLOW* to construct initial data for the full rotor solution. The command

../../tools/rotcgrid

performs a grid rotation to create the full rotor mesh file *ducted.meshrot* (this should be verified with **ls**). Next, the command

../../tools/rotcflow

performs a rotation of the steady state solution for the full rotor geometry and creates the file *ducted.restart.oldrot.* To launch the angle of attack solution, it is necessary to move the full rotor files just created into the standard mesh and restart file names. This may be accomplished by the following commands:

mv ducted.mesh ducted.mesh.steady

mv ducted.restart.old ducted.restart.old.steady

mv ducted.meshrot ducted.mesh

mv ducted.restart.oldrot ducted.restart.old

The first two commands above are intended to preserve the data for the steady flow calculation in case we wish to refer to it later. The second two commands simply place the full rotor mesh and restart data into the proper file names.

Now the angle of attack solution may be started with the command

../../src/aoa/aoa <ducted.unsteady.input >ducted.unsteady.output

This operation will likely take several hours on a workstation-level computer.

When the time-dependent solution is complete, an ls command reveals several other new files not previously encountered.

`ducted.p3dfr.1`   `ducted.p3dfr.2`   ...   `ducted.p3dfr.11`

The new files are all instantaneous *PLOT3D* absolute flow files generated at constant iteration intervals during the time-dependent calculation. This data may serve to illustrate the nature of the time-dependent flowfield during the course of the calculation.

A.5 Plotting the Output

At any time during the procedure outlined above, certain data may be plotted using the *ADPAC* tools plotting program *AOAPLOT* by issuing the command

../../tools/aoaplot

and following the menu instructions to generate the plots desired. The convergence history, power coefficient history, and per blade power coefficient history may be plotted using the *ADPAC* plotting program *FULLPLOT*. For example:

../../tools/fullplot <ducted.powercoef

The resulting file *fort.15* now contains *PostScript* output which may be previewed or sent directly to a *PostScript*-compatible printer for plotting.

This concludes the *ADAPC* User's Manual.

APPENDIX B. *ADPAC* DISTRIBUTION LIST

ADPAC DISTRIBUTION LIST

NASA Contract NAS3-25270
Task Order #2
User's Manual CR-187105
Final Report CR-187106

<u>GOVERNMENT AGENCIES:</u>

NASA Headquarters
600 Independence Avenue, SW
Washington, DC 20546

Attn: RJ/C. C. Rosen
RP/S. Wander
RP/J. R. Facey

NASA Lewis Research Center
21000 Brookpark Road
Cleveland, OH 44135

Attn:	J. J. Adamczyk	M.S. 5-9	(2 copies)
	C. L. Ball	M.S. 86-1	
	L. J. Bober	M.S. 77-6	
	D. R. Boldman	M.S. 86-7	
	B. Clark	M.S. 77-6	
	R. W. Claus	M.S. 142-5	
	J. H. Dittmar	M.S. 77-6	
	J. F. Groeneweg	M.S. 77-6	(2 copies)
	C. E. Hughes	M.S. 77-6	
	R. J. Jeracki	M.S. 77-6	
	C. M. Kim	M.S. 77-6	
	J. Lytle	M.S. AAC-1	

A. J. Mahajan	M.S. 23-3	
C. J. Miller	M.S. 77-6	(15 copies)
D. P. Miller	M.S. 77-6	
R. D. Moore	M.S. 77-6	
L. D. Nichols	M.S. 142-5	
C. W. Putt	M.S. 142-2	(2 copies)
D. R. Reddy	M.S. 5-11	
T. S. Reddy	M.S. 23-3	
R. Srivastava	M.S. 23-3	
G. L. Stefko	M.S. 23-3	
R. P. Woodward	M.S. 77-6	
J. A. Ziemianski	M.S. 86-1	
Report Control Office	M.S. 60-1	(4 copies)
Tech. Utilization Office	M.S. 7-3	
AFSC Liaison Office	M.S. 501-3	

NASA Ames Research Center
Moffett Field, CA 94035

Attn: Library — M.S. 202-3

NASA Langley Research Center
Hampton, VA 23665

Attn: F. Farassat	M.S. 461
M. H. Dunn	M.S. 904
Library	M.S. 185

NASA Scientific and Technical Information Facility
P.O. Box 8757
BWI Airport, MD 21240

Attn: Accession Dept. (6 copies and the FF427 form)

Sverdrup Technology, Inc.
2001 Aerospace Parkway
Brookpark, OH 44142

Attn: J. Bridges	M.S. 77-6
E. Envia	SVR-3
R. M. Nallasamy	SVR-3
S. M. Ramachandra	SVR-3
O. Yamamoto	SVR-3

ENGINE MANUFACTURERS

Allison Gas Turbine Division, GMC Corp.
P.O. Box 420
Indianapolis, IN 46206-0420

Attn: (Handled internally at AGT.)

General Electric Company
Aircraft Engine Group
1 Neumann Way
Evendale, OH 45215

Attn:	P. Gliebe	Mail Drop A-304
	C. Lenhardt	Mail Drop A-330
	M. Majjigi	Mail Drop A-319
	M. Pearson	Mail Drop A-317
	L. Smith	Mail Drop H-4
	C. Whitfield	Mail Drop A-304

Hamilton Standard Division - UTC
Windsor Locks, CT 06096

Attn:	D. B. Hanson	M.S. 1A-3-6
	F. B. Metzger	M.S. 1A-3-6

Pratt & Whitney Aircraft - UTC
Commercial Products Division
400 Main Street
East Hartford, CT 06108

Attn:	D. Hopwood	M.S. 162-07	(3 copies)
	D. Mathews	M.S. 165-11	
	W. Lord	M.S. 169-23	
	T. Wynosky	M.S. 169-23	

United Technologies Corporation Research Center
Silver Lane
E. Hartford, CT 06108

Attn:	M. Barnett	M.S. 20
	R. Davis	M.S. 20
	D. Dorney	M.S. 20

AIRFRAME MANUFACTURERS

Boeing Commercial Airplane Company (BCAC)
P.O. Box 3707
Seattle, WA 98124-2207

Attn:	B. W. Farquhar	M.S. 0L-22
	W-H. Jou	M.S. 7H-96
	R. Cuthbertson	M.S. 79-84

Douglas Aircraft Company Division
McDonnell Douglas Corporation
3855 Lakewood Blvd.
Long Beach, CA 90846

Attn:	M. Joshi	M.S. 36-60
	F. Lynch	M.S. 36-60
	G. Page	M.S. 35-86
	W. Siegele	M.S. 202-15

General Dynamics Convair
P.O. Box 80844
San Diego, CA 92138

Attn:	B. Bergman	M.Z. 36-1240
	S. Strande	M.Z. 55-6950
	K. Taylor	M.Z. 54-6890

UNIVERSITIES

Georgia Institute of Technology
School of Aerospace Engineering
Atlanta, GA 30332-0800

Attn: Dr. L. N. Sankar

Lockheed-California Company
P.O. Box 551
Burbank, CA 91520

Attn: Library

Ohio State University
Department of Aeronautical
and Astronautical Engineering
Columbus, OH 43220

Attn: Prof. G. M. Gregorek

Pennsylvania State University
Department of Aerospace Engineering
233 Hammond Building
University Park, PA 16802

Attn: Dr. B. Lakshminarayana

Purdue University
School of Aeronautics and Astronautics
W. Lafayette, IN 47907

Attn: Dr. J. P. Sullivan
Dr. M. Williams

Purdue University
School of Mechanical Engineering
W. Lafayette, IN 47907

Attn: Dr. S. Fleeter

Rohr Industries, Inc.
P.O. Box 878
Chula Vista, CA 92012-0878

Attn: Library

Texas A&M University
Aerospace Engineering Department
College Station, TX 77843-3141

Attn: Dr. K. D. Korkan

University of Arizona
Aerospace and Mechanical Engineering Department
Aero Building #16
Tucson, AZ 85721

Attn: Dr. E. J. Kerschen

University of Missouri-Rolla
Mechanical Engineering Department
Rolla, MO 65401-0249

Attn: Dr. Walter Eversman

Mississippi State University
Department of Aerospace Engineering
P.O. Box A
Mississippi State, MS 39762

Attn: Dr. D. Whitfield
Dr. M. Janus

NASA
National Aeronautics and Space Administration

Report Documentation Page

1. Report No. NASA CR-187105	2. Government Accession No.	3. Recipient's Catalog No.
4. Title and Subtitle Investigation of Advanced Counterrotation Blade Configuration Concepts for High Speed Turboprop Systems, Task II - Unsteady Ducted Propfan Analysis (Computer Program User's Manual)		5. Report Date May 1991
		6. Performing Organization Code
7. Author(s) Edward J. Hall Robert A. Delaney James L. Bettner		8. Performing Organization Report No.
		10. Work Unit No. 535-03-10
9. Performing Organization Name and Address Allison Gas Turbine Division General Motors Corporation P.O. Box 420 Indianapolis, IN 46206-0420		11. Contract or Grant No. NAS3-25270
12. Sponsoring Agency Name and Address National Aeronautics and Space Administration Lewis Research Center Cleveland, OH 44135-3191		13. Type of Report and Period Covered Contractor Report Final
		14. Sponsoring Agency Code

15. Supplementary Notes

Prepared in cooperation with NASA Project Manager Christopher J. Miller, NASA Lewis Research Center, Cleveland, OH

16. Abstract

The Primary objective of this study was the development of a time-dependent three-dimensional Euler/Navier-Stokes aerodynamic analysis to predict unsteady compressible transonic flows about ducted and unducted propfan propulsion systems at angle of attack. The computer codes resulting from this study are referred to as Advanced Ducted Propfan Analysis Codes (ADPAC). This report is intended to serve as a computer program user's manual for the ADPAC developed under Task II of NASA Contract NAS3-25270, Unsteady Ducted Propfan Analysis.

Aerodynamic calculations were based on a four-stage Runge-Kutta time-marching finite volume solution technique with added numerical dissipation. A time-accurate implicit residual smoothing operator was utilized for unsteady flow predictions. For unducted propfans, a single H-type grid was used to discretize each blade passage of the complete propeller. For ducted propfans, a coupled system of five grid blocks utilizing an embedded C-grid about the cowl leading edge was used to discretize each blade passage. Grid systems were generated by a combined algebraic/elliptic algorithm developed specifically for ducted propfans. Numerical calculations were compared with experimental data for both ducted and unducted propfan flows. The solution scheme demonstrated efficiency and accuracy comparable with other schemes of this class.

17. Key Words (Suggested by Author(s))		18. Distribution Statement
Euler Navier-Stokes Ducted Propfan	Unsteady ADPAC Turbofans	Unclassified-Unlimited

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of pages	22. Price
Unclassified	Unclassified	119	A06

NASA FORM 1626 OCT 86